AUTOMATIC CLASSIFICATION AND METADATA GENERATION FOR WORLD WIDE WEB RESOURCES

CHARLOTTE JENKINS BSc.

A thesis submitted in partial fulfilment of the
requirements of the University of Wolverhampton
for the degree of Doctor of Philosophy

November 2002

Signature.....Charlotte Jenkins.....

Date...Tuesday 12th November 2002

Acknowledgements

Special thanks to:

- Mike Jackson and Peter Burden for supervising the project
- Librarians from the University of Hull, Keith Donaldson Library, for their vital and greatly appreciated assistance
- My friends and colleagues at the University of Hull especially Dr Tanko Ishaya who has provided copious amounts of support, advice and inspiration
- My parents for encouraging me throughout

Abstract

The aims of this project are to investigate the possibility and potential of automatically classifying Web documents according to a traditional library classification scheme and to investigate the extent to which automatic classification can be used in automatic metadata generation on the web.

The Wolverhampton Web Library (WWLib) is a search engine that classifies UK Web pages according to Dewey Decimal Classification (DDC). This search engine is introduced as an example application that would benefit from an automatic classification component such as that described in the thesis. Different approaches to information resource discovery and resource description on the Web are reviewed, as are traditional Information Retrieval (IR) techniques relevant to resource discovery on the Web. The design, implementation and evaluation of an automatic classifier, that classifies Web pages according to DDC, is documented. The evaluation shows that automatic classification is possible and could be used to improve the performance of a search engine. This classifier is then extended to perform automatic metadata generation using the Resource Description Framework (RDF) and Dublin Core. A proposed RDF data model, schema and automatically generated RDF syntax are documented. Automatically generated RDF metadata describing a range of automatically classified documents is shown.

The research shows that automatic classification is possible and could potentially be used to enable context sensitive browsing in automated web search engines. The classifications could also be used in generating context sensitive metadata tailored specifically for the search engine domain.

Contents

1. Introduction

The World Wide Web is an unstructured, hyperlinked tangle of information. Pages can be added, deleted, linked and unlinked in a completely unregulated fashion. The volume of information available and the transient nature of that information make finding required material something of a challenge. Search engines and classified directories have become essential tools for locating information. Such tools enable users of the Web to enter a query or browse a classified list and derive a set of 'results' which are usually direct hyperlinks to pages thought to be relevant to the user's requirements. These tools have evolved from manually maintained 'hot lists' and directories like GENVL (McBryan, 1994), Galaxy and Yahoo, into fully automated search engines like Alta Vista, Lycos, Excite, Infoseek, HotBot etc (see figures 3 and 4 in chapter 2).

This introduction explores the evolution of search engines from Veronica to Google (see 1.1) and discusses the factors influencing that evolution. The merits of classification in tools for resource discovery and the trade off between manual classification and automatic index generation are discussed. The importance of metadata in relation to search engines is discussed. There is also an introduction to The Wolverhampton Web Library (WWLib) - a UK search engine that classifies Web pages according to Dewey Decimal Classification (DDC).

The predominant hypothesis of this project is that automatic classification according to a traditional library classification scheme is possible and can be used to assist in the acquisition of context sensitive metadata describing Web resources. There has, so far, been no reliable, consistent mechanism for automatically acquiring accurate, unbiased, up-to-date, standard compliant, interoperable, extensible, 'machine understandable' metadata for tools for resource discovery. The automatic classifier and metadata generator, designed and developed as a result of this project (discussed in detail in chapters 3, 4 and 5), address these issues and are intended to influence and enhance future development of WWLib.

1.1 The Evolution of Search Engines

Prior to the Web, the most popular method for retrieving information from the Internet was Gopher (Linder, 1992). Plain text, image and sound files were organised by category into hierarchical structures on Gopher servers which were then accessed using a Gopher client. Hierarchical menus and sub-menus led the user to the required information in an uninspiring but organised, logical fashion. Information was grouped by category and the items on each server were registered with the *Mother of all Gophers* (Linder, 1992) in Minnesota. A search mechanism known as *Veronica* (Foster and Barrie, 1993) could be used to interrogate the information held by the Mother Gopher and find the location of required information. The notion of maintaining a central resource in this manner, was appropriate before the Web when there was much less information available. Publishing information on Gopher required prior knowledge of certain configuration details which, along with the fact that it was visually unimpressive, prevented it from ever appealing to the masses. A comparatively small collection of unimaginatively formatted but reasonably well organised information soon gave way to a mass of multimedia, hyperlinked information with no central resource and no simple method for locating anything - the Web.

GENVL - Generate Virtual Library (McBryan, 1994) - was the first tool which could loosely be described as a directory, in the modern 'classified directory' sense. It attempted to emulate the logical structure of Gopher on the Web. GENVL was named the *Mother of all Bulletin Boards* because, like Gopher, it built a hierarchy of user-supplied virtual sub-libraries. The concept was the same as Gopher but GENVL did not have the monopoly that the *Mother of all Gophers* had. Pages that were not registered with GENVL were often equally as popular as pages that were. In fact, there were no laws of convention associated with the Web at all and the ease with which one Web page could provide hyperlinks to many others eliminated dependency on a single central register. The hierarchical, classified nature of GENVL, however, was exploited to the advantage of a number of later tools, most notably Yahoo.

Experience with GENVL showed that it was insufficient to rely entirely on user submission for resource discovery. In answer to this the World Wide Web Worm (McBryan, 1994) was developed -

the first automated Search Engine. It worked by using what is now commonly known as a *robot* or *spider* - in other words a mechanism for automatically retrieving documents from the Web and analysing them for embedded URLs (hyperlinks within the document that lead to other documents). When embedded URLs were found, those documents would be retrieved and analysed for further URLs and so on until the whole Web had been retrieved. A database was maintained that kept a record of each URL and where it was found. The World Wide Web Worm's user interface provided the means by which a query string could be entered into the input field of an HTML form which, when submitted, would be used to query the database. The results from such a query comprised a list of sites whose URL, title or heading fields were found to contain some or all of the terms in the query string. Additionally, each result was accompanied by the URL of the page in which the document was cited providing a citation index. The World Wide Web Worm won *The Best of the Web* award in 1994. The concept was taken on and improved by other mechanisms such as Lycos and Infoseek who developed more rigorous robots and more comprehensive text analysis techniques that maintained more comprehensive databases. Figure 1 shows the evolution of the main search engines.

Name	Date	Type
GENVL	1993	Classified directory
WWW Worm	1993	Primitive Search Engine
Galaxy	1993	Classified directory
Yahoo	1994	Classified directory
Lycos	May 1994	Search Engine
Infoseek	Early 1995	Search Engine
Excite	Late 1995	Search Engine
AltaVista	Dec 1995	Search Engine
HotBot	1996	Search Engine
Google	1998	Search Engine

Figure 1. Evolution of the main search engines and classified directories

In the mid 1990's (1994-1997) it seemed that new search engines, classified directories and metasearch engines (providing an interface for querying several search engines simultaneously) were appearing every month. They all suffered to some degree from the following drawbacks (Lindop et al 1997):

- Lots of irrelevant results not matching the user's query in any obvious fashion;
- Information overload - far too many results with interesting ones often being hidden among pages and pages of poor matches.
- Dead links - links to resources that no longer exist;
- Repetitive results, the same resource appearing several times over;
- US bias - most or all of the results pointing to resources in the US;
- Slow response times, especially in the afternoon (GMT) when transatlantic 'traffic' becomes particularly congested;

- ❖ Inconsistency of advanced query options between tools and the use of complex Boolean syntax - each tool requires queries to be expressed in a different format and this can often be confusing and misleading for the user especially those not accustomed to using Boolean algebra.

In the late 1990s the number of new search engines to emerge declined noticeably. It seems that not enough attention was paid to traditional Information Retrieval (IR) and librarian techniques. Although the Web is very different in many ways from traditional books, journals and the kind of corpuses used in testing traditional IR systems, it would appear that some of the same lessons can be learnt. The well established, well supported search engines (Alta Vista, Infoseek) and the few new ones to emerge (Google) seem more thoughtful in their approach to the problem. Search engines no longer just advertise the size of their corpus, as they did four years ago, instead they boast Natural Language Processing or Automatic Query Expansion or Probabilistic Retrieval - ideas well rooted in older research (see chapter 2). Users are understandably more impressed with accurate, well focused, up-to-date results than purely the capacity to acquire huge amounts of data, although obviously comprehensive coverage is also an admirable quality. The ability to structure information in an 'intelligent' fashion, once acquired, and consequently retrieve the appropriate information for any given query is what is sought.

1.2 The Merits of Classification

Classified directories provide access to manually classified documents that are clustered according to a pre-prescribed classification scheme. Automated search engines use a robot to retrieve documents from the Web, which are then automatically analysed and used to generate huge unclassified indexes. The advantage of automated tools is that they provide much more comprehensive Web coverage and are generally more up-to-date due to the continuous activity of their robots. The lack of classification and human intervention, however, appears to result in a tendency to overload users with irrelevant, poor quality results. In contrast, Yahoo has maintained its popularity (Lindop et al. 1997) as a manually maintained classified directory, because it provides very accurate, high quality information and it is intuitive to use. Classified tools usually enable users to browse a classification hierarchy where it is possible to focus their query on certain subject areas. Results are then restricted to those subject areas making the occurrence of irrelevant results and information overload very uncommon.

The advantages of document clustering and classification over keyword based indexes have been debated in Information Retrieval (IR) research for quite some time. Good (1958), Fairthorne (1961) and Salton (1968) discussed the merits of automatically and logically organising electronic documents into groups in the late 1950's and 1960's. The evolution of automated Web search engines from manually maintained lists and directories has further demonstrated the strengths and weaknesses of these two approaches.

Documents that share the same frequently occurring keywords and concepts are usually relevant to the same queries. Clustering such documents together enables them to be retrieved together more easily and helps to avoid the retrieval of irrelevant unrelated information. Classification usually enables the ability to browse through a hierarchy of logically organised information which is often considered a more intuitive process than constructing a query string. Keyword based indexes usually manage to find documents that contain specified keywords but find it difficult to simultaneously identify documents that share the same concepts. Indexes are however comparatively simple to construct automatically. Analysing documents for index terms is far easier than assigning a document to an appropriate classification group automatically. Consequently, classification is usually associated with human defined metadata or catalogue entries.

1.3 The Importance of Metadata

The acquisition of accurate metadata describing a resource is another complex issue that can hinder the performance of automated search engines. Metadata is data about data, in other words, information describing a resource that can be used to identify what the resource is, or what it is about, without having to actually analyse the data itself. Obviously metadata or resource description is important for search engines, it enables them to match resources with user queries. A typical example of some metadata describing a resource might be:

- the URI identifying the location of the resource;
- the title of the resource which can be used for identification;
- some keywords indicating subject matter and key topics;
- an abstract or summary - again indicating subject matter;
- the last modified date showing how up-to-date the resource is;
- a unique identifier assigned by the system to uniquely identify this resource and distinguish it from the others.

Critics of the Web and its architecture consider the lack of a reliable mechanism for resource description within the Web's architecture to be a huge oversight (this view was expressed by Ted Nelson, the inventor of hypertext, in his keynote address at the 7th international World Wide Web Conference in 1998). The Web without metadata has been likened to a city without signposts and street names with no names above the shop windows; if you manage to find the right shop you have to walk right in and look around before you know it's the right shop. Mechanisms have been introduced that enable authors to embed metadata such as keywords and summaries, in their pages which can then be picked out and used by applications such as search engines. Most search engines, however, ignore these 'meta tags' as they are open to abuse; some authors use meta tags to hide inappropriate information or 'spam' which they believe will improve their ranking in search engine results. It seems some authors are more concerned with the number of 'hits' their pages get than attracting the right audience for their material.

The World Wide Web Consortium (W3C, 1999) have introduced the Resource Description Framework (RDF) (Swick et al. 1998) which they hope will provide a platform for expressing interoperable yet extensible metadata element sets. W3C talk of a *Web of Trust* (Berners-Lee, 1997) where each individually accessible object on the Web is well described using RDF. The evolution of the *Web of Trust* requires comprehensive resource description which can only be achieved automatically. A method for automatically generating RDF metadata is introduced in chapter 5.

1.4 WWLib

The Wolverhampton Web Library (WWLib) is a World Wide Web search engine that classifies UK Web pages according to Dewey Decimal Classification (DDC)(OCLC Forest Press, 1999). The original version was developed in 1995 as a result of poor response times, US bias and 'information overload' from the big US search engines. The decision to use DDC evolved from the notion that library science – that has been responsible for organising vast amounts of information for decades – has a lot to offer the comparatively chaotic task of information resource discovery on the Web.

The original version of WWLib relied to a large degree on manual maintenance and as such can best be described as a classified directory that organised resources according to DDC. A new fully automated version is being designed and developed, WWLib TNG (The Next Generation), which will support a robot, automatic indexing and an automatic classifier. Figure 2 shows an outline design of the WWLib-TNG architecture. This diagram represents an initial draft of the proposed architecture, identifying the individual components required in implementing a fully automated web library. It in no way represents a finished design. A number of different projects will address different aspects of the individual components. This particular project addresses the issues involved in developing just one of the components - the classifier. Issues surrounding the integration of the individual components are largely beyond the scope of this project.

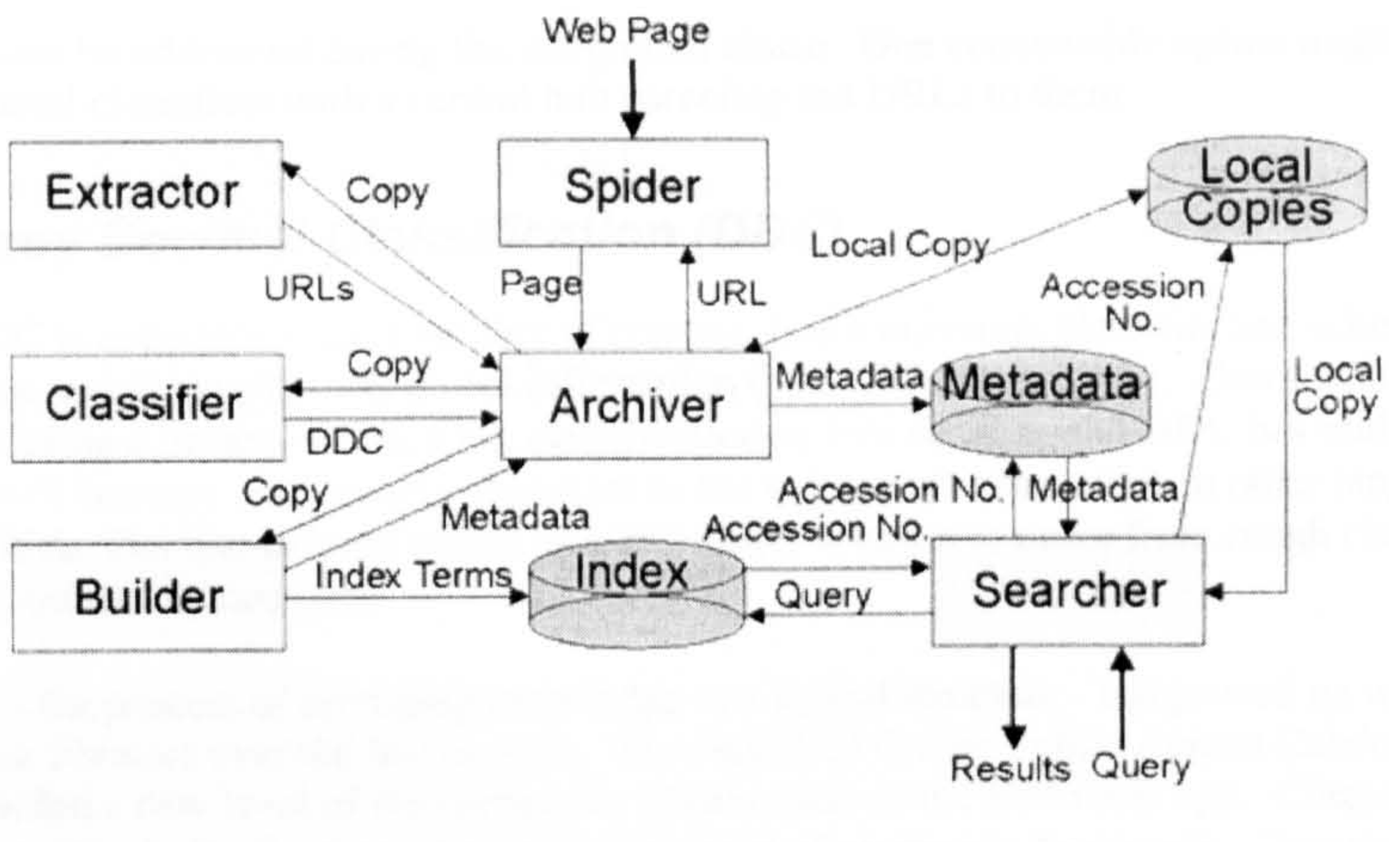


Figure 2. Outline design of the WWLib-TNG Architecture

There are essentially six components:

1. A Spider that automatically retrieves documents from the Web;

2. An Archiver that receives Web pages from the Spider, stores a local copy, assigns to it a unique accession number and generates a new metadata file. It also distributes local copies to the Extractor, Classifier and Builder and adds subsequent metadata generated by the Classifier and the Builder to the assigned metadata file;

3. An Extractor that analyses pages provided by the Archiver for embedded hyperlinks to other documents. If found, URLs are passed to the Archiver where they are evaluated to check that they are pointing to locations in the UK, before being passed to the Spider;

4. A Classifier that analyses pages provided by the Archiver and generates DDC classmarks;

5. A Builder that analyses pages provided by the Archiver and outputs metadata which is stored by the Archiver in the document's metadata file and is also used to build the index database that will be used to quickly associate keywords with document accession numbers;

6. A Searcher that accepts query strings from the user and uses them to interrogate the index database built by the Builder. It then uses the resulting accession numbers to retrieve the appropriate metadata and local document copies and then uses all this information to generate detailed results, ranked according to relevance to the original query.

The main goal of this particular project has been to develop the classifier, a tool which given a document, automatically classifies it according to DDC. Automatic classification has the potential to combine the advantages of classified directories with the advantages of automated search engines and result in an accurate, intuitive, comprehensive, classified search engine.

Clearly there are synchronisation issues regarding the speed at which the classifier can classify in comparison with the speed at which the spider, archiver and builder can locate, archive and index. The main aim of the classifier, however, is to combine automation with classification in a manner that offers an advantage over existing manual classification techniques. Existing solutions provided by other search engines, involve manually classifying a subset of the resources that have been automatically indexed. The automatic classifier component will work more quickly and more continuously than any human classifier. Although there may be some delay between indexing and classification, automatic classification offers a big advantage over current manual approaches. Resources can be registered (assigned a metadata file) as soon as they are indexed and that metadata file will be updated as they are classified. The manner in which the classification queue is processed is

an issue that must be addressed during the integration phase. One conceivable option might be to run a number of parallel classifiers with a central hub parceling out URLs to them.

1.5 Dewey Decimal Classification (DDC)

The use of DDC is appropriate for a number of reasons: It is a universal classification scheme covering all subject areas and geographically global information (Mai Chan et al. 1996). Users who are accustomed to using a library will find the classification system familiar and DDC has multilingual scope, which will become increasingly important as the volume of information in other languages grows on the Web. The hierarchical nature of DDC makes it easier to move from rough classifications to increasingly more accurate ones.

Classification - the process of arranging knowledge in a logical structure - has proved its worth in traditional book libraries over the last century. The classified Online Public Access Catalogues (OPACs) provided a new level of interaction for library users in the electronic age. Classification is being used increasingly for the organisation and retrieval of electronic documents. Devising a classification scheme from scratch is a time consuming and difficult task. There are several well known, universal classification schemes available, these include DDC, Library of Congress Classification (LCC) and the Universal Decimal Classification (UDC). Of these, DDC is the most well used globally (Mai Chan et al. 1996) and is particularly popular in the UK.

DDC was invented in 1873 and first published in 1876 by Melvil Dewey, then a student-assistant at the library of Amherst College. Dewey was born on the 10th October 1851 - by coincidence, the tenth day of the tenth month. In the 19th century classification in libraries was centred around available shelving - each time a particular classification out grew its shelf, the whole classification scheme had to be reorganised. This meant that classification schemes were different from one library to the next because they all depended on the physical environment. Dewey invented the idea of a general classification scheme that could be used in any library, where the books were assigned to an intellectual space within the classification scheme according to their subject matter, rather than to a fixed physical location. He used decimal fractions to denote the classifications and each book only needed to be classified once and could then be placed in the appropriate location in any library. This system enabled libraries to grow without having to reclassify all the books as they added new shelves.

Dewey divided the entire universe of knowledge into ten main classes, which were each divided into ten divisions, each of which had ten sections and so on until appropriately fine classifications were reached. Books were then mapped on to this knowledge hierarchy. Numbers 0 to 9 were used on each level of the hierarchy for example: 200 is the top level class Religion, 220 is the division for the Bible and 225 is the section for the New Testament.

Dewey is the most widely used classification scheme in the world. It is used in more than 135 countries and has been translated into more than 30 languages (Mai Chan et al. 1996). DDC is now in its 21st edition and is published in both print and electronic form (OCLC Forest Press, 1996).

1.6 Structure of this thesis

The rest of this thesis will be structured in the following chapters:

2. **Literature Review**
   This chapter includes a review of the available tools for information resource discovery on the web including search engines, classified directories, meta search engines, geographically specific and subject specific resources. There is also a review of Information Retrieval (IR) literature focussing on the implications for resource discovery on the web. The chapter ends with a review of metadata literature.

3. **Design of the classifier**
   This chapter covers the design of an automatic classifier that classifies UK Web pages according to DDC. The design is compared with that of an earlier classifier that was developed alongside the original WWLib (TOS). The classifier is an object oriented system and this chapter includes a

discussion of the necessary objects and classes including those used to build an object model of the DDC class hierarchy.

4. **Evaluation of the classifier**
This chapter describes an experiment where a corpus of 20000 URLs were classified and a randomly selected subset of those classifications were rated by librarians from the University of Hull Keith Donaldson Library who also provided a manual classification for each page. An analysis of the results is provided.

5. **Metadata Generation**
The classifier was configured to extract other useful metadata from a document. This chapter discusses which elements are extracted and why they are considered useful in the context of WWLib-TNG. An extension of Dublin Core for WWLib - Wolverhampton Core - is proposed and there is a discussion of the Resource Description Framework (RDF). An RDF data model and schema are generated for Wolverhampton Core and metadata is then automatically generated in RDF/XML syntax.

6. **Conclusions**
This final chapter reflects upon the thesis and defines the contribution. Issues requiring further research are identified.

2. Literature Review

This chapter provides an overview of relevant research literature and of existing tools and techniques for information retrieval and organisation on the Web. Section 2.1 describes the different categories of tools for information resource discovery. Section 2.2 provides an overview of Information Retrieval (IR) literature. Section 2.3 introduces a number of different formats and frameworks for expressing and exchanging metadata elements that have been developed for the Web domain. Section 2.4 introduces some significant automatic classification projects including the one documented in this thesis. Section 2.5 provides a summary of the findings in this chapter.

2.1 Tools for Information Resource Discovery on the WWW

2.1.1 Classified Directories

The hierarchical, classified nature of GENVL (McBryan 1994) was exploited to the advantage of a number of later tools, most notably Yahoo (see Figure 3). Yahoo is still respected as one of the best manually maintained classified tools for locating information (Lindop et al. 1997). It has its own proprietary classification scheme under which Web resources are grouped. Adding new pages to Yahoo is comparatively simple; an HTML form provides the means by which users simply select a category from Yahoo's pre-prescribed classification scheme. Data taken from users about each document is used to build a database. Users then wishing to locate required information are given the choice to either browse the classification hierarchy, refining their search from rough classifications to increasingly more accurate ones, or query the database with a query string. In either case this classified approach has the advantage of displaying clusters of documents that have been manually classified into the same category.

Early classified directories attempted to solve the problem of resource discovery by employing human "scouts" who spent their time browsing for new sites. The popularity of Yahoo became such that scouts were given the job of rating and reviewing user-supplied pages rather than looking for new material. Although later automated tools with their Web roaming robots and automatic indexing (see section 2.1.2) such as Lycos, Alta Vista and Excite looked as though they might leave the manually maintained directories behind, the merits of classification and human defined metadata have since re-emerged as very important issues and Yahoo has maintained its popularity.

Arguments for manually maintained classified directories (Lindop et al. 1997) centre around the notion that they provide quality rather than quantity. An appropriate category for each new resource is chosen by the user (often the author of the page) manually and keywords for indexing purposes are also entered manually. This human interaction combined with a well structured classification hierarchy usually ensures that users seeking information are not inundated with irrelevant, misleading suggestions, as they often are with automated, unclassified tools. The ability to browse the classification hierarchy is considered more intuitive for novice users (Lindop et al. 1997). Advanced users, who might find browsing frustrating, have the option to enter a query string.

The most damning criticism of Yahoo style classified directories is that, due to their manual data acquisition and maintenance, they suffer from poor Web coverage and out of date information (Lindop et al. 1997). In answer to such criticisms Netscape (1999) have introduced the Open Directory Project (see figure 3). The Open Directory Project demonstrates a different approach to maintaining a directory, its developers claim that search engines and classified directories alike are unable to cope with the volume of sites that have now emerged and that "link rot is setting in". A "vast army of volunteer editors" have been encouraged to sign up and help maintain the directory. The theory is that an army of editors is more likely to encourage comprehensive Web coverage and up-to-date links than a small editorial staff. Any Web user can go to the Web site, choose a topic they know something about and sign up as an editor for that subject area.

Another criticism of classified directories is that queries are matched only against the keywords, descriptions and titles entered by the user (or editor) and not the full text of the document which means relevant documents could be missed unless they happen to appear in the same category as those that are found. Classified tools that offer rating and reviewing as part of the service have been criticised for constraining query results and inducing biased information (Lindop et al. 1997), suggesting that too much human intervention is undesirable. It was in answer to criticisms such as these that automated search engines were first introduced.

Figure 3 shows a comparison of a number of classified directories.

Name	URL	Search Facility	Initial Categories	Site Reviews	Advanced Search
Open Directory Project	http://dmoz.org/	Y	15	Y	Y
Galaxy	http://www.einet.net/galaxy.html	Y	10	N	Y
Link Center	http://www.linkcenter.com/	N	20	N	N
Magellan	http://www.mckinley.com/	Y	18	Y	Y
Nerd World	http://www.nerdworld.com/	Y	12	Y	N
Search.com	http://www.search.com/	Y	14	N	N
Yahoo	http://www.yahoo.com/	Y	14	Y	Y

Figure 3 Classified Directories

2.1.2 Automated Search Engines

The concepts introduced by the World Wide Web Worm were taken on and improved by other more sophisticated search engines over the proceeding two years. Lycos, Infoseek, Excite and Alta Vista used full text indexing techniques and super-efficient robots to compile huge indexes (Sullivan. 1999).

Indexing the entire Web, even using a robot, is an impossible task. The Web is always changing, every minute new pages are added while old ones are changed or deleted. Robots are employed in revisiting known resources to detect changes, as well as discovering new ones, in an attempt to cope with this transient behaviour. "Dead links" - links to pages that no longer exist - are a problem for search engines and classified directories but they are noticeably more common in the results of classified directories because of their inability to perform automatic database updates. Some search engines maintain a record of how often sites update their information so that sites that are constantly changing are revisited more regularly.

There are certain codes of conduct governing the behaviour of robots. A text file called robots.txt placed in the root directory of any Web server can be used to specify areas of the server that may not be accessed by a robot. This is known as robot exclusion. Martijn Koster, also the author of the Aliweb search engine (Koster, 1994), was involved in the specification of a standard for robot exclusion which is now incorporated into the HTML 4.0 standard. He also maintains The Web Robots Pages (Koster 1997) at WebCrawler where the activity of all known robots is monitored. New robots can be registered on this site and there is a wealth of advice for robot writers. In addition to robot exclusion there are other conventions that robots should observe, such as allowing a time delay between requests to avoid "rapid-fire" on a server.

After pages have been discovered by a robot they must then be indexed. Classified directories obtain keywords for indexing purposes from the user when the document is submitted, search engines must index documents automatically. Most search engines boast full text indexing which means virtually every word in every document is matched against user queries. Although this means that relevant documents are rarely overlooked, it also means that irrelevant ones, that happen to contain certain relevant words out of context, can be retrieved if terms are not weighted according to significance (see section 2.2.1 on indexing). Alta Vista was very well received initially because of its very powerful and effective robot. Unfortunately, a good robot combined with full text indexing and a comparatively poor retrieval mechanism leads to high recall but low precision (see section 2.2), in other words information overload. Search engines have had to turn to more tried and tested IR indexing algorithms to deduce keywords that are particularly relevant to the subject of a document, resulting in more accurate indexing . According to Pedersen and Chang (1999) most search engines use some form of Vector Space Model and with IDF (Inverse Document Frequencies) and Boolean filtering (there is an explanation of these techniques in section 2.2).

Once documents have been indexed, information (metadata) needs to be stored about each resource. Typical metadata will include the title, URL, IP address, summary or description, keywords or index terms, file size, last modified date, the date the resource was first discovered, the date it was last checked for validation and so on. Although there are metadata standards (see section 2.3), most search

engines define their own proprietary formats. This information then needs to be stored in such a way that the retrieval mechanism has fast and easy access to the index terms.

The retrieval mechanism facilitates the identification and retrieval of documents relevant to user queries. Various approaches to the retrieval and ranking of results are utilised. Often the retrieval mechanism is heavily dependent on the indexing strategy (both these issues are discussed in section 2.2). Depending on the retrieval mechanism, advanced search options including Boolean syntax and/or phrase matching or natural language processing may be available to the user submitting a query.

The user interface plays an important role in obtaining a well focused query from the user and it is also important in the presentation of query results. The items resulting from a query are usually organised into some kind of rank order by the retrieval mechanism and are then presented to the user, a number at a time, with the most relevant appearing first. Metadata about each retrieved document is presented with the title appearing as a hyperlink to the document itself. The amount of metadata displayed varies from one search engine to the next depending on the information that is originally stored when indexing. It is important that results are clear and concise with well described items.

In summary, automated search engines generally comprise the following components:

- A robot that continually retrieves documents and analyses them for hyperlinks to other documents in an attempt to provide comprehensive Web coverage;
- An indexer that uses an IR indexing strategy to extract accurate index terms from the document;
- A database where metadata describing each resource is stored;
- A retrieval mechanism that takes user queries and quickly retrieves and ranks relevant documents from the database;
- A good user interface that encourages the user to input a coherent, well focused query and subsequently presents a clear set of results.

A number of automated search engines also offer a browsable classified directory, the entries of which are usually a subset of those found in the search engine database that are considered worthy of notice by staff who manually maintain the directory.

The main criticism of automated tools (Lindop et al. 1997) is that they tend to overload users with irrelevant, misleading results. Complex Boolean syntax is often required to focus queries appropriately which can be very confusing for novice users. Due to the lack of human intervention, results often contain links to very poor quality information and potentially useful information can be very badly indexed and described. The lack of classification can lead to documents that happen to share the same relevant words but not necessarily shared relevant context being displayed next to each other in the results.

Figure 4 shows a list of automated search engines with a comparison of available features.

Name	URL	Fully Indexed	Boolean Search	Classified Directory	Description	Relevance Score	Date	Proximity Search	META support
Alta Vista	http://www.altavista.digital.com/	Y	Y	Y	Y	N	Y	Y	Y
Excite	http://www.excite.com/	Y	Y	Y	Y	Y	N	Y	Y
Google	http://www.google.com/	Y	N	N	Y	N	N	Y	N
HotBot	http://www.hotbot.com/	Y	N	Y	Y	N	Y	N	Y
Infoseek	http://www.infoseek.com/	Y	N	Y	Y	Y	N	N	Y
Lycos	http://www.lycos.com/	N	Y	Y	Y	Y	N	Y	Y
OpenText	http://search.opentext.com/	Y	Y	N	Y	Y	N	Y	N
WebCrawler	http://webcrawler.com/	Y	Y	Y	Y	Y	N	N	Y

Figure 4. Automated Search Engines

2.1.3 Other Approaches

2.1.3.1 Meta Search Engines

Meta Search Engines provide the interface for querying the databases of a number of search engines and classified directories from the same page. The service provided by these tools varies considerably. Some provide a series of direct links to a large selection of search engines, others provide one input field and query a series of databases more transparently. Querying each search engine individually usually results in the user interacting with each search engine directly. Those that take one query string and submit it to several tools often post-process the results by collating and ranking them. This is perhaps a more useful service but the overhead involved is obviously considerable. The degree to which queries are pre-processed - translated into the correct syntax for each search engine - is not clear; in most circumstances complex Boolean queries are not advisable via meta search engines. Some users take advantage of the larger bandwidth of a local meta search engine to access remote resources (in the US). The number of databases queried varies from one meta engine to the next with some just querying the most prevalent - Alta Vista, Lycos, Infoseek, Excite - and others querying a longer and more varied list of search engines.

Figure 5 lists some of the available meta search engines.

NAME	URL	RESULTS COLLATED?	NUMBER OF SEARCH ENGINES QUERIED
All 4 One	http://all4one.com/	N	4
Cyber411	http://cyber411.com/	Y	6
Dogpile	http://www.dogpile.com/	Y	14
Highway61	http://www.highway61.com/	Y	4
Internet Sleuth	http://www.isleuth.com/	Y	6
MetaCrawler	http://www.metacrawler.com	Y	6
Metasearch	http://metasearch.com/	N	6
Pro Fusion	http://www.designlab.ukans.edu/profusion/	Y	6
Savvy Search	http://guaraldi.cs.colostate.edu:2000/form	Y	11
StartingPoint	http://www.stpt.com/	N	160

Figure 5 Meta Search Engines

2.1.3.2 Geographically Specific Resources

Most major search engines and classified directories such as Alta Vista, Excite, Lycos, HotBot Infoseek, Google, Yahoo, Galaxy, Magellan... and so on, are situated in the USA. Internet users in other parts of the world often have problems with this due to poor response times, particularly in the afternoon (GMT) when the US are awake and transatlantic traffic becomes exceptionally congested. A tendency to provide US biased information can also be a problem. US bias came about, not only because most of the major search engines are located in the US, but also because initially the US were making more extensive use of the Web than most other places. US information had a tendency to drown out most other information, simply because there was more of it.

A growing number of local search engines emerged in the UK, Europe and other parts of the world that provide information on the local domain. Some people believe it is more beneficial to mirror the big US engines locally than to keep reinventing the wheel by developing more and more new search engines. It may be that the well established search engines have better resources and can therefore afford to provide a better service with bigger, faster machines and faster, higher bandwidth connections. The disadvantage of mirroring, however, is that although the response time problem is solved, the database in most cases remains the same - US biased results with US biased reviews.

Figure 6 shows some of the many geographically specific resources.

NAME	URL	COUNTRY	TYPE
ANANZI	http://www.anazi.co.za/	South Africa	Search Engine
ANZWERS	http://www.anzwers.com.au/	Australia & New Zealand	Search Engine
Channel Hong Kong	http://www.chkg.com/	Hong Kong	Search Engine
Euroferret	http:/www.euroferret.com/	Europe	Search Engine
Kolibri	http://www.kolibri.de/	Germany	Search Engine
Search.NL	http://www.search.NL/	Holland	Search Engine
Swiss Search	http://search.ch/	Switzerland	Search Engine
TechnoFind	http://www2.technofind.com.sg/tf/	Singapore	Search Engine
UK Index	http://www.ukindex.co.uk/uksearch.html	UK	Classified Directory
UK Plus	http://www.ukplus.co.uk/	UK	Classified Directory
UKSearch	http://www.uksearch.com/	UK	Search Engine
The UK Web Pages	http://www.neosoft.com/_dlgates/uk /ukgeneral.html	UK	Classified Directory
YELL	http://www.yell.co.uk/	UK	Classified Directory
ZZZ	http://www.zzz.ee/otsi/index_en.html	Estonia	Search Engine

Figure 6. Geographically Specific Resources

2.1.3.3 Subject Specific Resources

Even the most comprehensive automated search engines cover just a small proportion of the total amount of information available - just 16% according to Lawrence and Giles (1999). Covering the Web in its entirety is an impossible task due to its constantly changing nature. One proposed method for improving coverage, at the same time as solving other problems such as information overload, poor quality information and irrelevant query results, is to provide a series of directories that are each dedicated to a specific subject area. Each directory is maintained by experts in the particular subject area who provide site reviews and ratings and ensure that accurate, high quality information is maintained in a well structured hierarchy.

This concept has been taken up quite seriously by a number of specialist groups. Resource Organisation And Discovery in Subject-based Services (ROADS) (Kirriemuir 1996) has encouraged the development of a number of high quality information gateways.

Figure 7 lists some subject specific gateways.

Name	URL	Subject	Type	Location
1.2.1.2.	http://www.1212.com/	Music	Classified Directory	France
Achoo	http://www.achoo.com/	Healthcare	Classified Directory	Canada
ADAM	http://www.adam.co.uk/	Architecture, Design And Media	Classified Directory	UK
ASE	http://www.uni-karlsruche.de /~un9v/atm/ase.html	Airport Search Engine	Search Engine	Germany
BizAds Business locator	http://bizads.2cowherd.net/	Businesses	Search Engine	USA
CampSearch	http://www.campsearch.com/	Summer Camps	Search Engine	USA
Computer ESP	http://www.uvision.com /search.html	Computer Companies and Products	Classified Directory	USA
EEVL	http://eevl.ac.uk/	Edinburgh Engineering Virtual Library	Classified Directory	UK
1st Global Directory	http://www.123link.com/	Business Products and Services	Classified Directory	USA
Motherload	http://www.cosmix.com /motherload/	Web Directories and Search Engines!	Classified Directory	USA
NetMall	http://www.netmall.com/	Goods and Services	Classified Directory	USA
OMNI*	http://omni.ac.uk/	Organising Medical Networked information	Classified Directory	UK
SHAREWARE. COM	http://www.shareware.com/	Software	Search Engine	USA
SOSIG	http://sosig.esrc.bris.ac.uk/	Social Sciences Information gateway	Classified Directory	UK
Sports Directory	http://www.sport-hq.com/	Sport	Classified Directory	USA

Figure 7. Subject Specific Resources

2.2 Information Retrieval

Evaluation of tools for information retrieval is usually based on two measures - recall and precision. Recall refers to the percentage of all relevant documents that are retrieved from a database and precision refers to the percentage of the documents retrieved that are relevant. For example, if documents on Medieval English Literature were sought from a database that contained 80 documents relevant to this query, 20 of which were retrieved along with 30 irrelevant ones - 50 documents being returned in total - recall and precision would be calculated as follows (Salton 1983):

$$\text{Recall} = \frac{\text{Number of items retrieved that are relevant}}{\text{Total number of relevant documents in the database}} = \frac{20}{80} = 0.25$$

$$\text{Precision} = \frac{\text{Number of items retrieved that are relevant}}{\text{Total number of documents retrieved}} = \frac{20}{50} = 0.4$$

There are two areas of search engine functionality that prescribe the degree of recall and precision; the indexing strategy and the retrieval mechanism.

2.2.1 Indexing

IR indexing strategies have evolved from the manual task of library cataloguing where librarians would manually specify a number of keywords to identify each item (book, journal, etc.). The performance of a search engine, in terms of recall and precision, relies heavily on its indexing strategy. Crucial issues here are what information is extracted from each document and how accessible that data then is.

There are generally two types of automatically generated index; weighted and unweighted (Kowalski 1997). In an unweighted index each term is stored with a value describing its location and little or no further information. These indexes best support Boolean searches where a document is either relevant or it is not. No indication as to the degree of relevance can be easily obtained from this kind of index.

With a corpus the size of the Web it is obviously advantageous to organise results from a query into a ranked list with documents that are likely to be most relevant at the top. In a weighted index terms are assigned a weight according to their frequency within the document. Luhn, Brookstein, Klein and Raita's theories all support the notion that the significance of a word, in terms of its power to reveal concepts within a document, is directly proportional to the frequency with which it occurs within the document (Kowalski 1997). Weight values assigned to index terms are commonly normalised to a figure between zero and one, one indicating the highest significance. The number of occurrences of the term in the database as a whole is often used to avoid common 'stop' words (eg. it, at, on, the, if, when, and, then, that... etc) being assigned a significant weight value and also to encourage more unusual words to acquire a heavier weight. This Inverse Document Frequency (IDF) technique is known to be used by most major search engines (Pedersen and Chang, 1999). Weighting the terms in this way enables the retrieval mechanism to score and rank documents according to their relevance to the user query. Often query terms are themselves weighted to identify the most important words in terms of their power to retrieve relevant documents. This is done by assigning weights according to word frequencies within the database.

The Vector Space Model (Salton, Wong and Yang, 1974) is a common IR approach to weighted indexing and subsequent retrieval. Documents are represented as vectors, each of which have a vector position for every known term (word) in the database. The indexing mechanism assigns a weight to the position of each found term depending on its frequency. Terms that are not found have a value of zero. Queries are then also translated into vectors so that a measure of similarity between the query vector and the document vector can be obtained. A variant of this approach is known to be used by the Excite search engine as part of its Intelligent Concept Extraction (ICE) process (Excite Inc. 1996).

Another common IR approach is based on a probabilistic model, the most common of which is known as the Bayesian Model (Kowalski 1997; Oddy 1981), whereby the probability of a document containing a particular concept is calculated on the basis that it contains certain words.

Some search engines claim to use natural language processing. This is where constructs within the language are identified; semantic information is combined with statistical information to identify phrases and word patterns. The frequent co-occurrence of terms across a range of documents is also used to identify phrases and concepts.

It is important that once the indexing terms have been ascertained, they are stored in a manner that enables fast access by the retrieval mechanism. A common method of quickly associating query terms with document accession numbers is to use an inverted file index. This is where every possible term has an entry in an index file with a list of associated document accession numbers. These accession numbers can then be used to look up further metadata and often a local copy of the full text of the document.

2.2.2 Retrieval

Retrieval algorithms used by the searching component of search engines generally fall into one of three areas; Boolean, probabilistic or natural language processing.

2.2.2.1 Boolean Searches

Many search engines encourage the use of Boolean syntax within user queries. A user wishing to locate documents about 'Equine Anatomy' might find that merely typing the two search terms into the query input box of some search engines leads to results referencing hundreds of pages about horses, but not anatomy, and/or hundreds of pages about anatomy, but not horses, drowning out the few relevant pages that are actually about equine anatomy. The query string 'equine AND anatomy' (capitalisation of operators seems to be common but not all search engines use this syntax) would probably be far more successful as only those documents containing both terms would be retrieved. The Boolean operators AND, OR and NOT are implemented using intersection, union and difference procedures from set theory.

Boolean logic provides a means for focusing queries well and can help to improve recall and/or precision. Sometimes it is possible to include parentheses to dictate the order of operators. For example, if a user wanted to retrieve documents about reptiles and/or mammals but not humans they could use the query 'reptile OR (mammal NOT human)'.

2.2.2.2 Fuzzy Boolean

When a query string contains more than one term, in the absence of any Boolean operators, Fuzzy Boolean (Excite Inc. 1996) is often used. Documents are ranked according to the number of terms matched. This tends to improve precision at the top of the list.

The term *Fuzzy Searching* (Kowalski 1997) is used to describe a mechanism that is often used as a result of very poor recall. Terms that have similar spelling to the query terms are sought in the assumption that the query terms have been incorrectly spelt.

2.2.2.3 Proximity searching and phrase matching

Some tools provide a mechanism for specifying that the search terms entered must appear adjacent to each other. This is usually indicated by the adjacency operator, ADJ. Proximity searches may also be encountered that specify that terms must occur 'near' each other. It is generally considered that two terms occurring near to each other give better indication of concept (Kowalski 1997). For example, a document containing the term 'historical' close to the term 'architecture' is more likely to contain information about historical buildings than a document that has these terms several paragraphs apart.

Proximity searches are usually based on the proximity of just two terms. If a user wanted to search for a whole phrase, it is often possible to enclose a phrase such as 'Child Development Psychology' in inverted commas, only those documents containing the exact phrase should then be retrieved.

2.2.2.4 Thesaurus searches and query expansion

One method of improving recall is to retrieve documents that, not only contain the query terms, but synonyms of those terms also. Electronic thesauri are available to enable this process. The problem with this approach is that often the focus of queries can be badly skewed by unsuitable synonyms resulting in improved recall but disastrously low precision. To avoid this negative result search engines supporting this feature often present the user with a list of synonyms relating to their original terms so that they can select relevant ones.

Statistical thesauri provide an alternative method. Instead of looking up semantic synonyms, terms that have a statistically high coincidence with the user's query terms within documents are added to the query. This approach is known as automatic query expansion. Often the original query is processed to reveal which terms are most likely to focus the query - those that are less common within the database - and the query is then expanded with statistical synonyms of those terms. The Muscat (Muscat 1997) search engine, EuroFerret (see figure 6) uses probabilistic retrieval in conjunction with 'Relevance Feedback'. This enables the user to indicate which results are most relevant to their query and similar documents with a high coincidence of significant terms are then sought.

2.2.2.5 Stemming and term masking

The retrieval mechanism may also improve recall by carrying out suffix and gerund stripping or 'stemming' on the query string. This means that any terms ending in "s", "ed", "ing", "ology",

"ologist", "ological" etc. will be stripped so that, for example, a search for "Psychological Conferences" will find a document containing the words "Psychology Conference" highly relevant.

Stemming is often used to improve recall but it can have a negative effect on precision. The Porter stemming algorithm (Porter 1980) identifies words with certain suffixes and replaces them with stemmed versions. This can result in decreased precision, as Kowalski (1997) points out 'memorial' and 'memorise' have very different meanings but would both be reduced to 'memory' by the Porter algorithm. An alternative method is to use a dictionary based approach such as Kstem (Kowalski 1997) where more accurate stems are obtained by replacing the word with the most appropriate stem obtained from a dictionary. Frakes' (1992) evaluation of stemming experiments confirmed that stemming algorithms only have a positive effect on recall, not on precision.

A different approach altogether is to use term masking (Kowalski 1997) in the query. The endings of words are masked and any combination of characters after the unmasked characters can be accepted as a match. For example the masked term psycho* could be matched against the terms psycho, psychology, psychologist, psychological and so on.

2.3 Metadata

Metadata is information describing a piece of information - data about data or resource description. As discussed in the Introduction (section 1.3), metadata is very important for applications on the Web and for search engines in particular. Search Engines use resource descriptions to match relevant resources with user queries. It is probable that most commercial search engines use their own proprietary format for resource descriptions. Many academic projects, however, particularly the Subject Specific Gateways described in the previous section (2.1.3.3) are interested in sharing resource descriptions to aid comprehensive Web coverage. In answer to this, metadata standards have been introduced, the most important being Dublin Core (see 2.3.2).

2.3.1 IAFA Templates

Probably the first metadata standard commonly used on the internet was the IAFA (Internet Anonymous FTP Archive) template (Beckett 1995). The IAFA template comprised a number of name-value pairs (now known as metadata elements) which could be used to describe a resource. As the name suggests, this template was first used for describing items belonging to an anonymous FTP archive. However, systems wishing to describe Web resources in a standard fashion soon adopted the IAFA template in the absence of an alternative. Examples of such resources are ALIWEB (Archie Like Indexing on the Web) and ROADS (Resource Organisation And Discovery in Subject -based services).

2.3.2 Dublin Core

The Dublin Core metadata element set (Dublin Core 1999) emerged from the digital libraries community to answer the need for a digital equivalent of a MARC record for networked resources. The Dublin Core Workshop series began in 1995. The purpose of the workshops was to define a 'core' set of metadata elements that would adequately describe any (textual or non-textual) networked resource and to enable proper metadata interoperability between tools for resource discovery. The first workshop (Weibel et al, 1995) was supported by the Online Computer Library Centre (OCLC of Dublin Ohio) and the National Centre for Supercomputing Applications (NCSA). The workshop resulted in the identification of 13 Dublin Core metadata elements:

1. **Subject**: The topic addressed by the work
2. **Title**: The name of the object
3. **Author**: The person(s) primarily responsible for the intellectual content of the object
4. **Publisher**: The agent or agency responsible for making the object available
5. **OtherAgent**: The person(s), such as editors and transcribers, who have made other significant intellectual contributions to the work
6. **Date**: The date of publication
7. **ObjectType**: The genre of the object, such as novel, poem, or dictionary

8. **Form**: The data representation of the object, such as Postscript file or Windows executable file
9. **Identifier**: String or number used to uniquely identify the object
10. **Relation**: Relationship to other objects
11. **Source**: Objects, either print or electronic, from which this object is derived, if applicable
12. **Language**: Language of the intellectual content
13. **Coverage**: The spatial locations and temporal durations characteristic of the object

The second workshop (Dempsey and Weibel, 1996) was held at Warwick University in the UK a year later and was supported by OCLC and the UK Office for Library and information Networking (UKOLN). This workshop resulted in the definition of a framework in which Dublin Core metadata elements could be expressed - the Warwick Framework.

The third workshop (Weibel and Miller, 1997) extended the number of core elements to 15 with the addition of :

14. **Description**: A textual description of the content of the resource
15. **Rights Management**: A link to a copyright notice or a rights management statement.

At the fourth workshop (Weibel, Iannella and Cathro 1997), the notion of Dublin Core qualifiers was introduced. Although the fifteen elements are considered suitable to provide a simple description of most document-like objects, it is often necessary to repeat the same element. These different values may need to be differentiated from each other and this can be achieved using qualifiers. For example the Subject element has a 'scheme' qualifier which can be used to differentiate a Subject element comprising a list of keywords from a Subject element comprising a classification classmark.

The fifth workshop (Weibel and Hakala, 1998) in Helsinki drew a line under development and specification of the semantics for unqualified Dublin Core. The "Finnish finish" formed the basis of the first formal standardisation of the fifteen Dublin Core elements. The notion of developing a formal data model for Dublin Core, for use with RDF was introduced.

The sixth and seventh workshops (Miller 1998, Scholz and Effelsberg 1999) discussed in more depth the Dublin Core data model and the refinement of a standard set of qualifiers.

2.3.3 The Resource Description Framework (RDF)

RDF has been introduced by W3C to provide a platform for expressing extensible yet interoperable metadata element sets within the Web's architecture. Using RDF it is possible to express metadata elements in Dublin Core or any other metadata format while maintaining interoperability. RDF schemas are used to define new or extended metadata element sets, these schemas form the RDF type system and are referenced from the RDF description of a resource using the namespace mechanism from XML. A detailed discussion of RDF can be found in chapter 5.

2.4 Automatic Classification

IR approaches to automatic classification involve teaching systems to recognise documents belonging to particular classification groups. This can be done by manually classifying a set of documents and then presenting them to the system as examples of documents that belong to each classification. The system then builds class representatives each of which consists of common terms occurring in the documents known to belong to a particular classification group. When the system subsequently encounters new documents it measures the similarity between the document and the class representatives. Each time a new document is classified it is used to modify the class representative to include its most commonly occurring keywords.

2.4.1 TAPER

The Taxonomy And Path Enhanced Retrieval (TAPER) system (Chakrabarti et al. 1997) uses techniques rooted in traditional IR to classify documents according to a hierarchical classification scheme. IR techniques are used to extract signatures from documents based on significant terms and these are then compared with signatures representing each node of the classification hierarchy. Each node has a different context specific stop word list that is applied to the document signature as it is

filtered down through the hierarchy. When a user queries the TAPER system, they are initially presented with a list of topic paths, rather than documents, this helps to focus the query to the most relevant areas of the classification hierarchy where subsequent relevant documents will be clustered.

2.4.2 Scorpion

Another project using DDC for automatic classification is the Scorpion project (Thompson et al. 1997) of OCLC. Their system combines library science with IR techniques to enable automatic subject assignment using DDC as a knowledge base. Documents are used as queries to a database of manually defined DDC information. The result from such a query identifies the subject matter of the document. The manually defined DDC information is maintained by OCLC using an electronic Editorial Support System (ESS). Scorpion uses ESS records to build its knowledge base. These are the same records that are used to produce the printed version of DDC and the DDC 21 CD ROM Dewey for Windows.

2.4.3 ACE

The Automatic Classification Engine (ACE) that has been developed to form part of the new fully automated WWLib TNG (see section 1.4) has similarities with both TAPER and Scorpion. Like TAPER it works by recursively filtering documents through a hierarchy of class representatives (keywords and synonyms), each one representing a node of the classification hierarchy, until appropriate leaf nodes are reached. However, like Scorpion, a manually defined vocabulary is used in the class representatives to accurately describe each node of the DDC classification hierarchy.

The classification process is used to acquire other useful metadata elements turning ACE into an Automatic Metadata Generator. The acquired metadata is generated in RDF syntax, using an RDF Schema to define a new metadata element set - the Wolverhampton Core - which is an extension of Dublin Core.

2.5 Summary

Different tools for resource discovery on the Web offer varying degrees of automation and manual classification. Although many implement automatic indexing, few so far offer automatic classification. Clearly there are benefits to the user associated with context sensitive classification. IR literature has greatly influenced the development of automatic indexing tools on the Web. Traditional approaches to automatic classification, however, have not been prevalent on the Web so far. A number of different metadata formats have evolved for the Web domain. RDF and XML provide a foundation on which interoperable, extensible metadata element sets can be defined and exchanged. This will be explored more in chapter 5. The next chapter documents the design of ACE, an automatic classification engine that will classify Web documents according to DDC.

3. The Design of the Automatic Classification Engine (ACE)

The Wolverhampton Web Library (WWLib) is a search engine that organises UK Web pages according to DDC. Each time a document is added to WWLib, an appropriate DDC classmark is assigned which should reflect the position of the document within the classification hierarchy according to its information content. The original WWLib - known as WWLib-TOS (The Original Software) - required manual classification of every document. This worked well when the rate at which new resources emerged was comparatively low. As new sites began to emerge more and more rapidly, the impossibly slow and tedious nature of manual classification prompted investigation into the possibility of automatically classifying new documents.

Alongside the original WWLib-TOS, a somewhat primitive automatic classifier was developed in 1994. This original classifier has become known as Old ACE. One of the key aims of this project has been to develop 'New ACE'. The two classifiers work in very different ways in terms of their treatment of the DDC class hierarchy. Old ACE takes a 'bottom-up' approach, matching the document with leaf nodes on the first parse, whereas New ACE takes a more traditional librarian 'Top-down' approach, starting at the top of the DDC hierarchy, with the classes shown in figure 8, and proceeding to investigate the subclasses of those classes where keyword matches are apparent. Old ACE uses just the text taken from the DDC classmark label with perhaps one or two added synonyms to represent each class. New ACE uses a much more detailed list of synonyms and keywords to represent each class, using the classmark label purely for identification.

000	Generalities
100	Philosophy, paranormal phenomena, psychology
200	Religion
300	Social sciences
400	Language
500	Natural sciences and mathematics
600	Technology (Applied sciences)
700	The arts, Fine and decorative arts
800	Literature (Belles-lettres) and rhetoric
900	Geography, history, and auxiliary disciplines

Figure 8. The ten top classes of the DDC classification hierarchy

The following sections provide a more detailed description of the two classifiers. Section 3.1 describes the operation of Old ACE. Section 3.2 describes the operation of New ACE. Section 3.3 comprises a detailed design of New ACE describing the attributes and behaviours of each object used in the implementation of the system. Section 3.4 provides a summary of this chapter.

3.1 Old ACE

In order to provide the facility to navigate the classification hierarchy and classified catalogue in WWLib TOS, a file was created which contained the textual labels and Dewey Decimal Codes for each DDC class with occasionally some added terms and synonyms, as follows:

```
629.1 Aerospace Engineering. Aircraft. Aeroplanes.
629.2 Motorised Land Vehicle Engineering.
629.22 Types of vehicles.
629.222 Cars
629.2223 Vehicles for public transportation.
629.22232 Taxis and Limousines.
629.22233 Buses.
629.22234 Ambulances.
629.223 Light trucks and lorries.
629.225 Work vehicles. Bulldozers and tractors.
```

This file will be referred to here as the DDC file (within the WWLib project this file has often been referred to as the thesaurus but this is rather a misnomer). Since the manual classification of Web pages

for WWLib-TOS had become an exceedingly tedious task, investigation soon began into the extent to which text matching between the actual pages and the DDC classmark label, found in the DDC file, could help in automatically classifying the pages.

3.1.1 Basic Strategy

In Old ACE each word in a document was checked against the complete DDC file and, if the word was present in the label of a particular class, the score for that class was incremented. When the entire page had been processed, the class with the highest score was determined and its Dewey Decimal Code used as the classification for the document. The incremental score associated with the recognition of a particular word was determined by a number of factors that are described in the following subsections. One of the interesting features of Old ACE was that it attempted phrase matching - words were not just recognised in isolation but in the context of adjacent words both in the DDC file and the document.

3.1.2 Web Page Parsing

The document was first broken down into a stream of words for the DDC file comparison mechanism. At any time the comparison mechanism would be working with the "current" word and the context provided by the previous ten input words. The buffer of ten words was known as the running buffer and was used for phrase matching. Only "left context" was used so that phrase matching was sensitive to the order of words in a phrase.

HTML tags, attributes and values were largely ignored except in circumstances where certain important tags were recognised. Typically words found within <title> .... </title>, <h1> .... </h1> had higher scores (i.e. were considered more significant) than words in other HTML contexts.

In recognising words, punctuation symbols were deleted, however commas and full-stops had a special effect. When the input stream crossed a comma or a stop, a number of blank words were placed in the running buffer, this reduced the possibility and significance of a word pair match across sentence or phrase boundaries.

Common words, as determined from a manually maintained file, were ignored. Words of two or fewer letters were ignored as were numbers and sequences of non-alphabetic characters such as e-mail addresses etc.

The only form of stemming applied to the document was depluralisation, i.e. the conversion of the plural form of a word to a singular. Other forms of stemming (e.g. Porter, 1980) were not applied to the document because the risk of forming non-words by incorrect application was thought to be too high in the context of the uncontrolled vocabulary of the WWW.

3.1.3 Internal representation of the DDC file

The external form of the DDC file was not suitable for fast searching therefore the first task for Old ACE was to build an internal representation of the file. In the course of doing so, a full set of stemming operations were applied to the DDC file labels, this was acceptable as the DDC file was locally generated and could, in theory, be manually tuned, as could the stemming mechanism, to avoid any problems with non-word generation.

3.1.4 Basic Operation

Old ACE searched the internal representation of the DDC file for each word found in the document. This was done using a binary search. If the word was not found, it was simply flagged as an unknown word and no further action was taken. If the word did occur in the DDC file then ACE determined the value or score to be associated with the word, which was the sum of the number of classes in which the word occurred plus an HTML context sensitivity factor. There were three HTML associated weights: h_title, h_h1 and h_h2 depending on whether the word was encountered within <title> ... </title>, <h1> ... </h1> or <h2> ... </h2> tags.

Each time a word match occurred, Old Ace worked backwards through the running buffer and the words associated with the current DDC file entry, from the synchronisation point, looking for another match. If such a match was found then a word pair in the document had also been found in the DDC

file, possibly with a different number of intervening words. When this occurred, the score was incremented in accordance with the rules shown in figure 9.

Rule 1	Both words adjacent	Multiply score by factor f1
Rule 2	Same distance from synchronisation point in input stream and thesaurus class record	Multiply score by f2 divided by the word distance from the synchronisation point
Rule 3	Different distances from synchronisation point	Multiply score by f3 divided by the sum of the two word distances from the synchronisation point

Figure 9. Rules associated with scoring matched words according to proximity

After the complete text of the page had been processed in this manner, a resulting score was associated with each class whose DDC file entry had words in common with the text of the page. It was then an easy matter to determine which class had the highest score and output the associated Dewey decimal code.

This Old ACE classifier achieved around 30% accuracy. Although far from acceptable, taking into account the manner in which it had been implemented, this was considered to be an encouraging initial experiment.

3.2 New ACE

An important aspect of this particular project has been to redesign and implement an improved classifier for WWLib-TNG, taking into account other research in the field and addressing some of the inadequacies of the original classifier. It was always the intention to redevelop WWLib-TNG with a distributed architecture (see figure 2.) and consequently the new classifier (new ACE) was implemented independently of the other components (many of which are the subject of other projects and publications). Most of the other components of WWLib-TNG were not yet available as the new classifier was developed. New ACE works as a stand-alone application that, given a URL or path to a local file, will automatically classify any web page according to DDC. The remainder of this chapter will concentrate on the design of this stand-alone web page classifier.

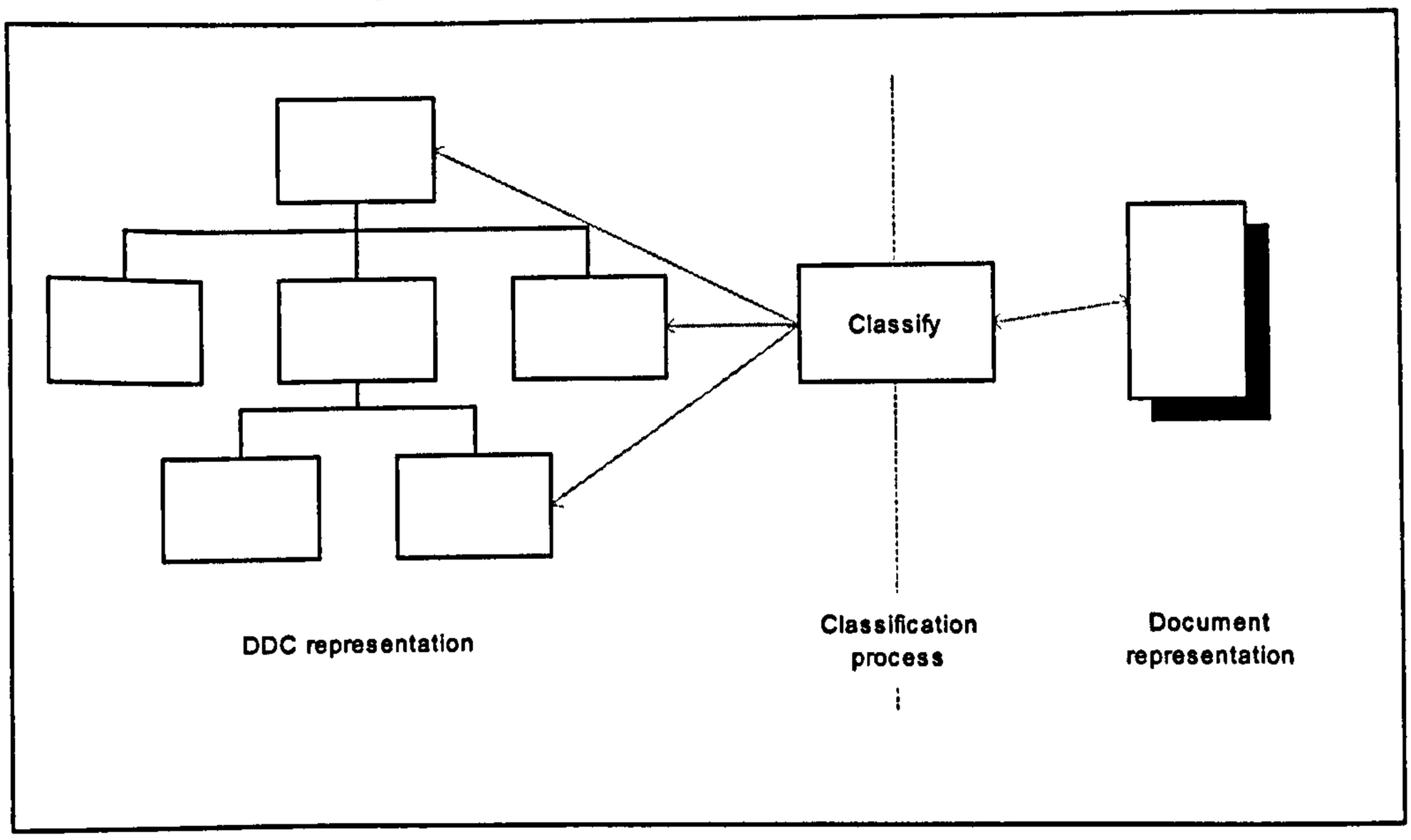


Figure 10. Classifier Architecture

The new classifier is written in Java and has an object-oriented design. As such, the first stage in the design process was to model the real world objects involved in the classification process. Most obviously the highest-level objects are those representing the Dewey Decimal Classification Scheme and that representing a document to be classified. These two entities came to represent quite distinct

'sides' of the architecture of the classifier as shown in figure10. The objects representing the DDC classification scheme and the object representing a document are quite independent of each other but are tied together using a 'classify' object which recursively compares the document object with the hierarchical objects representing the DDC classification scheme. The components of the classifier and the representations stored within are discussed over the following sub-sections.

3.2.1 Representation of the Dewey Decimal Classification Scheme

Dewey Decimal is a hierarchical classification scheme. Each 'node' of the hierarchy represents an intellectual space in which, historically, books have been classified and library shelves have been organised accordingly. Each node can have a maximum of ten sub-nodes. The hierarchy provides a navigational map, to aid resource discovery, but it does not provide an accurate representation of the relationship between different fields of human knowledge. A subject area with a greater number of sub-topics will spread further down the hierarchy rather than across it due to the decimal limitation. This characteristic means that automatically comparing the relationship between different sets of classmarks is an almost impossible task as will be discussed in the evaluation chapter.

Each node of the hierarchy can be represented by the following attributes:

- A unique classmark comprising the decimal code and an accompanying textual label
- A series of keywords and synonyms that reflect the subject of this node
- A maximum of ten sub-nodes representing sub-topics

The second of the above attributes was a key area of focus during the design of the new classifier. This series of keywords, known as a class representative, was modelled in the new classifier using a vector of keyword objects. The classmark and sub-nodes were also modelled using separate objects as described in the following sub-sections.

3.2.1.1 Class Representatives

Relying purely on the terms from the DDC textual labels to identify the classification of a whole document, as Old ACE did, has clear deficiencies. There are many examples of DDC classes where the textual label does not properly define the class or where the label makes no sense without the context of the superclass (examples are "604 Special topics" and "408 Kinds of persons treatment"). Most automatic classifiers documented in IR literature (van Rijsbergen, 1981) use the concept of a class representative, which is a set of terms and synonyms used to define a particular class or node of the classification scheme. New ACE provides a class representative comprising a list of manually defined terms and synonyms for each DDC class.

The terms and synonyms were derived from terms in the textual label and, where appropriate, synonyms of those terms and other terms found associated with that class in the Dewey for Windows CD ROM (OCLC Forest Press, 1996). The manual definition of the DDC class representatives was a time-consuming process but was considered a worthwhile exercise since each class would only need to be defined once enabling long-term accurate automation at the cost of short-term manual input. A team of experts for each subject area would be required to fully define the complete set of class representatives. An acceptable covering of all but one of the ten main classes was generated for the purposes of this prototype, over 120 classes in total.

In traditional IR classifiers, the class representatives are often automatically generated using pre-classified training documents. Training was not adopted here because of the unreliable nature of web documents; as Wong and Fu (2000) suggest "The quality of the resulting [classifier] highly depends on the fitness of the training examples" and "traditional feature extraction methods are not suitable in the web domain" because "a Web page usually contains a small number of words and most words appear only one or two times." Their statistical analysis of the web domain shows that most web documents contain less than 500 words and most words in a web document will rarely appear more than twice. Traditional IR classifiers are designed to deal with much wordier documents and are often designed to work in restricted subject domains such as that described by Moens and Dumortier (2000) in their article on " Text Categorization: The Assignment of Subject Descriptors to Magazine Articles". It was considered that automatic training would not be suitable for precisely defining the DDC representation. A useful feature might have been the ability to automatically identify commonly occurring terms in documents belonging to a particular classification during the classification process and for those then to

be presented as possible extensions to the class representative. The approach adopted, however, was similar to that used by the Scorpion project (Thompson et al., 1997) of the On-line Computer Library Centre (OCLC) where the manually defined ESS records that are used to generate the definitive DDC documentation, are used as the knowledgebase for automatic subject assignment. Such records could conceivably be used to generate an industrial strength version of New ACE.

Each class representative is stored as a vector of keyword objects within an instance of the 'Dewey' object (described in section 3.2.1.3). The keyword object (described in section 3.2.2.3) comprises the actual word and an associated weight. Words within the class representatives are undifferentiated by weight for this prototype but this facility could be used to weight more significant words within the class representatives or to adjust weight according to position within the hierarchy. Following some experimentation, a set weight of 10 was assigned to each word in the class representatives. Appendix A documents an experiment where a series of pages were classified using different set weights on the class representative terms. 10 was found to be the most effective. This weight assists significant word matches to be identified between documents and class representatives that cover wide subject areas with large vocabularies. This weighting clearly had a positive effect on the hierarchical filtering process described in section 3.2.3.2. and on the significance test described in section 3.2.3.4.

3.2.1.2 The Classmark Object

The classmark object has two key attributes; one representing the DDC code and another representing the textual label. The object also provides methods for comparing two classmarks and for setting and retrieving a score associated with a classmark.

In the context of the DDC representation, classmark objects are used within instances of the 'Dewey' object purely to identify which node of the hierarchy that instance represents. During the classification process, if a match is found between a document representation and an instance of the 'Dewey' class that represents a leaf node, the associated classmark object is scored and copied to the document representation (described in more detail in section 3.2.2.4).

3.2.1.3 The Abstract Class Dewey

In order to model the hierarchical structure of DDC, an abstract class 'Dewey' was defined which all classes representing nodes within the hierarchy extend (inherit). This class ties together the two attributes described in the previous subsections – i.e. the class representative and the classmark object – with a vector of instances of itself representing the next layer of the hierarchy beneath the current node. Figure 11 shows the basic structure inherited by all DDC objects extending the abstract class 'Dewey'.

A separate class was written for every implemented DDC class (127 classes). Each of these classes extends abstract class Dewey creating its own class representative comprising appropriate keywords and adding its own list of subclasses representing the next layer of the hierarchy unless it is a leaf node in which case there are no subclasses. Instances of these classes are instantiated dynamically as required in the classification process (see section 3.2.3). In retrospect Dewey could have been implemented as a concrete class and each DDC class could have been created from a text representation as an instance of the Dewey class. In some respects the text representations would have been as complicated to produce as the independent classes. Since they are all instances of the Dewey class by inheritance there are no interoperability issues and they are in fact treated polymorphically as instances of class Dewey. It would be simple to develop an application that would generate text representations from the separate DDC classes if required.

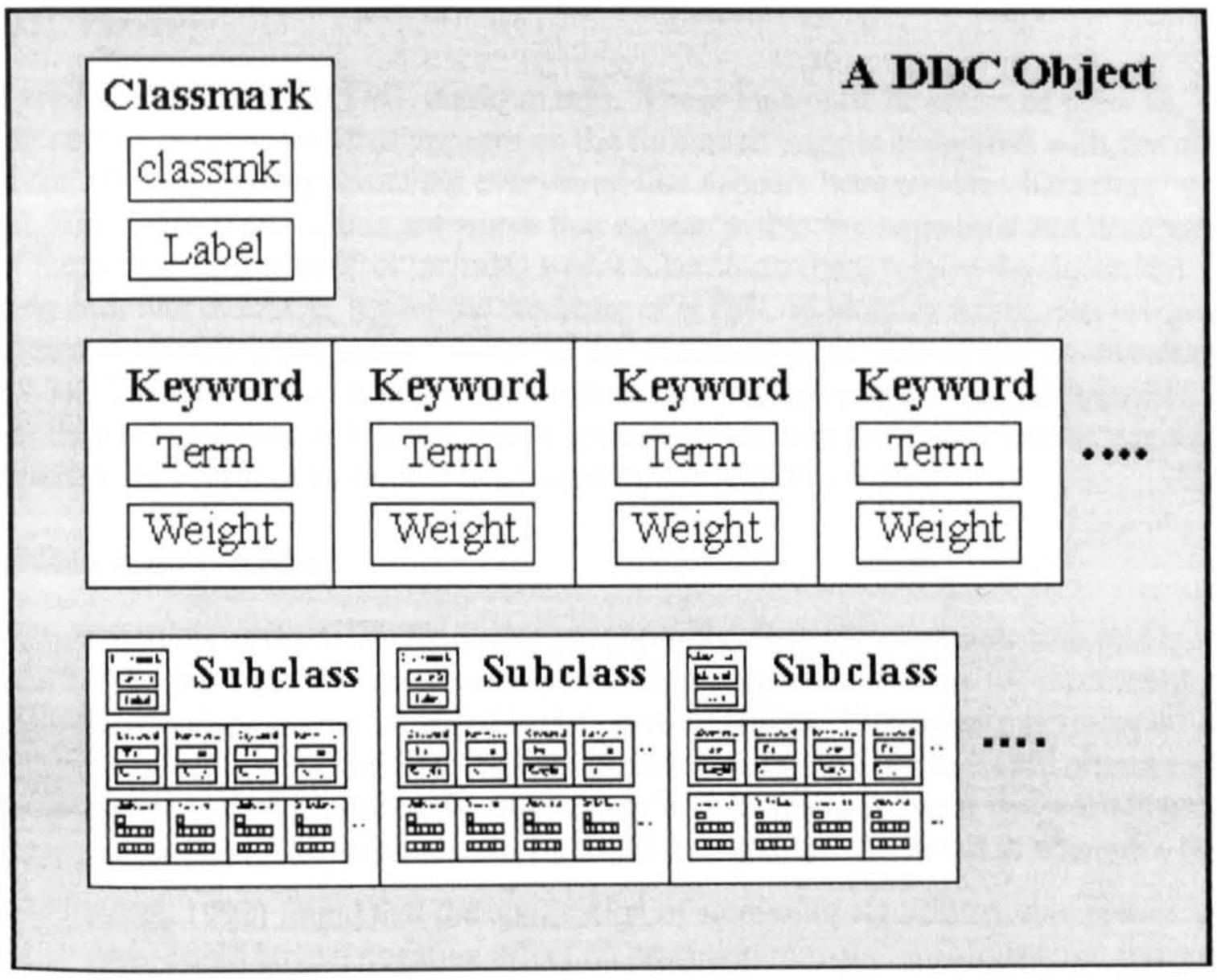


Figure 11. Basic Dewey structure inherited by all DDC objects

3.2.2 Document Representation

In order to classify documents they must be compared with the class representatives of the DDC hierarchy. It is necessary to first acquire a representation of the document – a document object – which comprises a list of keyword objects, identical in structure to the class representatives that represent each DDC class. The vector of keywords representing the document is acquired as follows:

```
open a connection to the document
read a line
while (the line is not null)
{
   remove unwanted HTML tags
   tokenise the line on white space and punctuation
   while (there are more tokens)
   {
      get the next token
      if (the token is not a stop word)
         add the token to a vector of keywords
   }
   read the next line
}
close the connection
```

The following subsections describe the processes involved in creating the document representation in more detail.

3.2.2.1 HTML Parsing

Web pages are formatted using HTML mark-up tags. These tags must be removed prior to classification so that only the text that appears on the formatted page is compared with the class representatives. This is done by removing everything that appears between the characters '<' and '>' in the HTML file. Exceptions to this are words that appear within the keywords and description attributes of meta tags. The classifier includes words taken from meta tags in the document representation and, like old ACE, it uses the structure of HTML to identify words that are potentially more significant in revealing the subject matter of the document (this feature will be discussed in section 3.2.2.3). The locations of certain tags are therefore marked using a special sequence of characters as they are removed so that the words appearing between those tags can be weighted accordingly when the vector of keywords is generated (see section 3.2.2.3).

3.2.2.2 Stopping and Stemming

The class representatives of the DDC hierarchy are naturally stopped. Common stop words such as "a", "at", "the", "if", "no", etc. are not generally considered significant terms for representing a DDC class and consequently they are not included in the representatives. Documents are stopped during the HTML parsing phase using a special Stop object. Before generating each keyword object each term found in the document is first passed to the Stop object which checks to see if the word is present in a list of common stop words. The stop list used by the Stop object can be found in appendix B.

Frakes' thesis (Frakes, 1992) found that the application of stemming algorithms, that reduce words to their canonical form, could have a negative effect on precision in tools for information retrieval (as discussed in section 2.2.2.5). A full stemming algorithm was therefore not implemented. Sub-string matching is equally dangerous because some words can be found within quite unrelated words. The word "ion" for example, - meaning an electrically charged atom or group of atoms – is the sub-string of every word that has the "tion" suffix (information, citation, creation, objection etc.). Consequently it was decided that terms in the class representatives would not be restricted to their canonical form. The same word with a number of different suffixes may occur in a class representative and each version is treated as a different word.

3.2.2.3 The Keyword Object and Keyword Weighting

As each new term in the document is identified it is used to generate an instance of a keyword object. Keyword objects are stored in a vector. Each object in the vector represents a unique word within the document. The keyword object has two attributes:

- Term – the actual word found in the document
- Weight – a weight associated with the term, which is used to indicate the significance of the term within the document. The frequency of the term is added to other weights associated with the location of the term as described below.

"Luhn, Brookstein, Klein and Raita's theories all support the notion that the significance of a word, in terms of its power to reveal concepts within a document, is directly proportional to the frequency with which it occurs within the document" (Kowalski, 1997). For this reason words within document representatives are often weighted according to their frequency. However recent research into Information Retrieval on the web has revealed that web documents do not have quite the same properties as traditional documents. As previously discussed in section 3.2.1.1, Wong and Fu (2000) found that most Web documents contain less than 500 words and most words in web documents rarely appear more than twice. Soderland (1997) and Hodgson (2001) have supported the notion that the structure of HTML can be used in identifying terms of particular importance. This was also found to be the case during early experiments with Old ACE (Burden and Wallis, 1996). The approach adopted for New ACE (like Old ACE) was to combine the two forms of weighting so that the frequency of the term is added to any additional special score deduced from its appearance within particular HTML tags. Previous experimentation with Old ACE (Burden and Wallis, 1996) had shown that the <TITLE> and <H1> tags were particularly useful in deducing words of particular significance. Figure 12 shows the tags that were used to identify significant words acquiring extra weight.

Tag	Extra weight acquired
<TITLE>...</TITLE>	10
<H1>...</H1>	10
<H2>...</H2>	5
<META NAME=KEYWORDS CONTENT=...>	10
<META NAME=DESCRIPTION CONTENT=...>	10

Figure 12. Html tags used to identify significant words that acquire additional weight

Having identified the tags considered important, it was then necessary to decide upon the additional weight added to words appearing within those tags. Often in Information Retrieval, weights are deduced using the term frequency and inverse document frequency. Because the classifier was designed to work as a stand-alone application that could be utilised independently of WWLib, the inverse document frequency of words was not available. The classifier has no memory of the terms appearing in previously classified documents. When weighting according to appearance within a tag the scoring necessarily becomes comparatively arbitrary. It is not uncommon in information retrieval systems for arbitrary weights to be set which are systematically adjusted during testing. Paice for example, in his 1993 SIGIR paper , (Paice and Jones, 1993), used an integer between 1 and 10 to weight constructs within documents. The weights were "initially assigned by pure guesswork" but were progressively adjusted during successive trials. After some experimentation, the weights shown in figure 12 were determined for weighting HTML constructs within the document representatives of New ACE. Appendix C shows the results of an experiment where a series of sample pages were classified using a number of different weight combinations on the HTML constructs. The combination shown in figure 12 was found to be the most effective.

Figure 13 shows a simple example HTML page and the vector keyword objects that would be generated to represent this page:

<HTML>

<HEAD><TITLE> Hillside Animal Sanctuary</TITLE></HEAD>

<BODY>

<H1>Welcome to Hillside Animal Sanctuary</H1>

<P>Our sanctuary provides a good home to all kinds of animals – dogs, cats, horses, rabbits, cows, sheep and pigs.</P>

</BODY>

</HTML>

Hillside	Animal	Sanctuary	Welcome	Hillside	Animal	Sanctuary	sanctuary	home	kinds	animals	dogs	cats	horses	rabbits	cows	sheep	pigs
10	10	10	10	10	10	10	1	1	1	1	1	1	1	1	1	1	1

Figure 13. A vector of weighted keyword objects representing a simple example document

3.2.2.4 The Document Object

The document object provides the methods required for maintaining and accessing the vector of weighted keyword objects described above and also ties in a vector of classmarks. Classmark objects are assigned to this vector as a result of the classification process described in section 3.2.3. Figure 14 shows the document object:

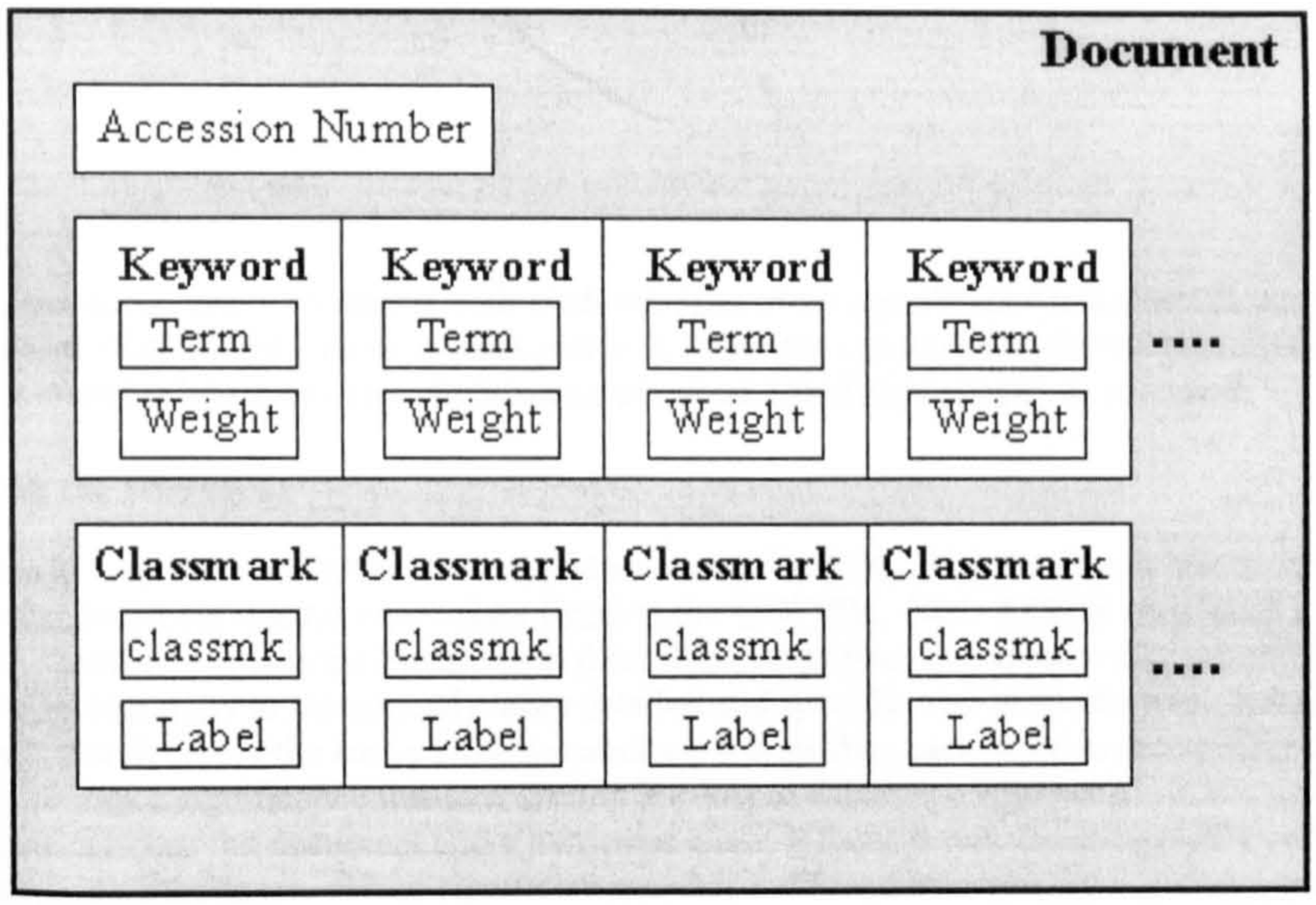


Figure 14. The Document Object

The Accession Number attribute represents a unique identifier used to represent and differentiate each document within WWLib. When the classifier is in stand-alone mode this attribute is unset.

3.2.3. Comparing Document representations with DDC nodes

The actual classification process involves comparing the document representatives described in section 3.2.2 with the class representatives of the DDC hierarchy described in section 3.2.1. This is a recursive comparison process co-ordinated by the Classify object (as indicated in figure 10). Initially the document representative is compared with the DDC objects representing the top ten DDC classifications listed in figure 8. Each time a significant match is found between the document and a DDC object, instances of the ten subclasses are dynamically instantiated and the document is then recursively compared with those classes and any significant subclasses.

3.2.3.1 The Classify Object

The classify object is passed an instance of the document object, representing a particular document to be classified. It uses a recursive method to compare this document representative with the class representatives of the DDC hierarchy. The recursive method is called proceed and operates as follows:

```
proceed (takes a Dewey object called ddc as a parameter)
{
   compare the document with the current ddc object
   if (there is a significant match between them)
   {
      if (ddc has no subclasses)
      {
         assign the classmark from ddc to the document
      }
      else
      {
         while (ddc has more subclasses)
         {
            proceed (with the next subclass from ddc)
```

```
            }
        }
    }
}
```

This means the document is compared with each subclass when significant word matches are found with the current DDC object. When a weak match is found the comparison process proceeds no further through that branch of the hierarchy. Subclasses are instantiated dynamically as required.

3.2.3.2 Using the Hierarchy

Old ACE made very little use of the hierarchical nature of the DDC classification scheme. It ran each word from the document against every class listed in the DDC file. New ACE is very much a hierarchical classifier - it uses the hierarchy to filter documents from broad class representatives at the top of the DDC hierarchy to increasingly more detailed and specific ones at the bottom. Instead of doing a word search across the entire DDC vocabulary, it considers each word in the context of its DDC class. It uses a significance test (see section 3.2.3.4) to determine whether there are significant word matches between the document and a particular class; if there is not, the comparison process does not proceed to any subclasses. When significant matches are found between the document and a leaf node, the document is assigned the classmark. Several classmarks could be assigned which, in theory, should all be appropriate to varying degrees. The classifier is configured to present only the highest scoring classmarks if there are several.

New ACE uses a top-down hierarchical approach, which is quite common in IR literature (van Rijsbergen 1981, Chakrabarti et al. 1997). The top-down approach better models the activity of a human librarian who always starts with the top classes and works down when assigning a classification to a new book in a library.

The TAPER system developed by Chakrabarti et al. (1997) at IBM Almaden Research Center uses customised stop lists to filter documents at each node of the classification hierarchy. New ACE achieves a similar filtering effect using customised class representatives at each node. To illustrate this, with a very simple example, figure 15 shows some of the subclasses of the 000 Generalities class.

This demonstrates how the classification hierarchy can aid the classification process. Documents matched against broad lists of keywords are filtered through sub-classes with more detailed, focussed terms. Ambiguous terms can be 'hidden' until a point where they can be considered in context. Words that would not normally be particularly significant in identifying subject matter such as 'learning' become more useful when considered in context – in this case in the context of Artificial Intelligence within Special Computer Methods. Words like 'computing' indicate the general subject area among broad categories, then as the classification process proceeds through the hierarchy more context specific terms and acronyms such as AI and intelligence are used to identify subject matter within computing in this case.

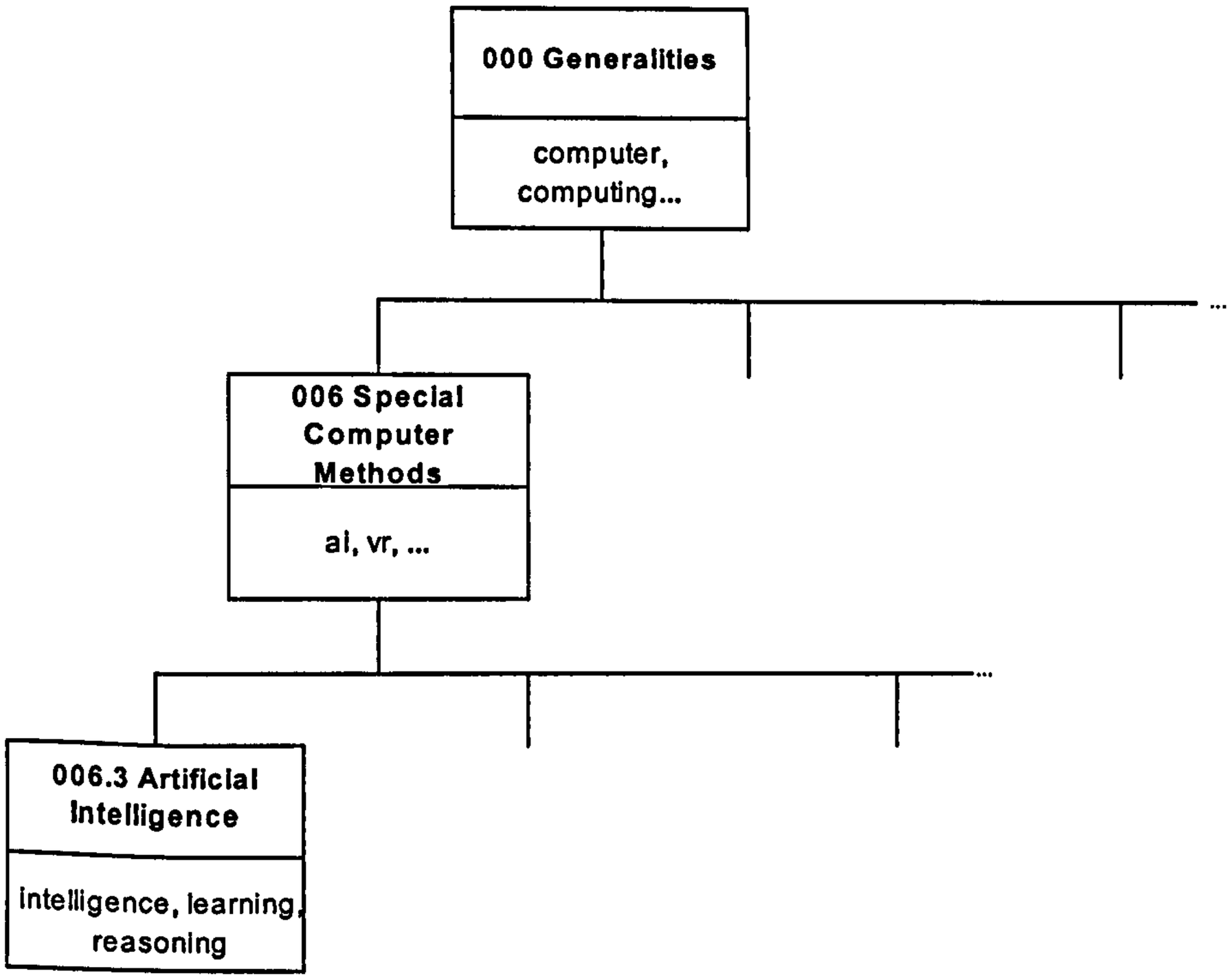


Figure 15. Showing some classes and class representatives of three levels of the Generalities sub-hierarchy

3.2.3.3. Scoring

A score for each document/DDC comparison is calculated by the score method on the Classify object as follows:

```
score (takes an instance of a Dewey object called ddc as a parameter)
{
   while (ddc has more keywords in its class representative)
   {
      get the next ddc keyword
      while (the document object has more keywords)
      {
         get the next document keyword
         if (the document keyword matches the ddc keyword)
         {
            add the associated weights to a total score
         }
      }
      reset the document keywords
   }
   return the total score
}
```

3.2.3.4 Significance testing

When New ACE compares a document with the class representative of a particular DDC class a score is obtained as a result of the comparison process as described above. The score is calculated by adding together the associated weights of each matched word. Before proceeding to compare the document with the subclasses of the current DDC class it works out if this score is significant. If it is not the comparison process will not proceed to the subclasses but instead will move across to the next class adjacent to the current one (or up and across if there are no more adjacent classes) in a typical recursive manner.

Initially, scores were simply totalled and the highest scoring classes selected as the appropriate classifications. However, longer wordier class representatives had an unfair advantage over shorter ones. There was an obvious inconsistency in the performance over long documents and shorter ones. It soon became apparent that some kind of formula that would normalise the lengths of the representatives was required. Measures of similarity were investigated such as the Dice Coefficient (van Rijsbergen, 1981) shown in figure 16:

$$2\frac{|X \cap Y|}{|X + Y|}$$

Figure 16. The Dice Coefficient

The Dice Coefficient is interesting in that it takes the intersection between two sets (which could be perceived as the total score in New ACE) and divides this by the sum of the lengths of the two sets. The integration of length normalisation was appealing. However there are a number of problems with Dice:

1. Sets do not normally contain repeating values so word frequencies cannot be taken into account
2. There are no weights of any kind incorporated into the Dice formula so the weights assigned according to position within certain HTML tags discussed in section 3.2.2.3 cannot be used with Dice
3. Dice is a similarity measure where two like sets are compared resulting in a value between 0 and 1. The two entities compared by New ACE are not like – one is a document, the other a representative of a DDC class that could comprise words taken from a wide range of subject areas in the case of top-level classes.

There follows an example of the application of Dice which highlights these problems:

The simple example HTML page discussed in section 3.2.2.3 and represented in figure 13 is classified by New ACE under 590 Animals. This indicates that the class acquires a significant match with both the 500 Natural Sciences class and the subclass 590 Animals. Taking, firstly the intersection between this document and the 500 (Natural Sciences) class, there are five words in common: animal, animals, dogs, cats and horses. The document has 17 words and the DDC class has 93 words so the Dice Coefficient is calculated as follows:

$$= 2\frac{5}{17+93}$$

$$= \frac{10}{110}$$

$$= 0.09$$

Similarity measures like Dice return a value between 0 and 1 where 0 represents no similarity and 1 represents a comparison between two identical sets. Based on this it is apparent that a result of 0.09 is not very helpful since this class should be flagged as significant. The problem here is that the range of terms covering unrelated aspects of Natural Sciences are drowning out significant ones concerning animals.

The formula performs more promisingly when the weights assigned to terms in the document representative according to frequency and occurrence within certain HTML structures (as discussed in section 3.2.2.3) are used to "overload" the top of the formula as shown below:

The words from the document title that appear in the intersection are as follows:

Word	Animal
Frequency	1
Weight	10

The words from the document first level headings (<H1>) that appear in the intersection are as follows:

Word	Animal
Frequency	1
Weight	10

The other words that appear in the intersection are as follows:

Word	animals	dogs	cats	horses
Frequency	1	1	1	1
Weight	1	1	1	1

If the weights are added to the formula it could be calculated as follows:

$$= 2\frac{10(1)+10(1)+1(4)}{17+93}$$

Note that since word frequencies are now incorporated, the word 'animal' is counted twice; once when it appears in the title (and acquires a weight of 10) and again when it appears in the first level heading (where again it acquires a weight of 10).

$$= 2\frac{10+10+4}{17+93}$$

$$= \frac{48}{110} = 0.436$$

This is clearly an improved score. It must be observed, however, that this can no longer be considered a measure of similarity since it is conceivable that a comparison could now acquire a result that is greater than 1 due to the overloading of the formula. Clearly there is never a case when two sets can be said to be anything more than the same. As discussed above the two entities under comparison are never likely to be identical (because one is a document, the other a class representative) so it seems reasonable to consider this formula as a significance test rather than a similarity measure. The classifier never asks the question "How similar are these two entities?" instead it asks the question "Are there significant word matches between these two entities?". It does this by setting a threshold - any result above the set threshold is considered significant.

Although the "Dice with weights" formula represents an improvement as far as the operation of New ACE is concerned, 0.436 is still not a particularly high score considering that this class should be flagged as significant due to the subclass dedicated to animals. Appendix D shows the results of a simple experiment where a number of documents were classified using different thresholds for the significance test. Resulting from this the "significance threshold" was set at 0.5. Clearly the result of 0.436 would still fall short of that threshold. It was discovered that incorporating weights acquired from the class representative in addition to those acquired from the Document improved the ability to flag significant matches. This meant that the threshold need not be set dangerously low where documents with some words out of context could get significant matches with inappropriate classes. Below is a detailed description of the significance test formula that was eventually derived that incorporates further weighting.

The formula that was devised is a modified version of the Dice Coefficient enabling representative lengths to be normalised while still considering word frequencies and HTML related weights. Unlike Dice, the new formula is not intended to be a measure of similarity since the two entities being compared are not like (one is a document and the other a class representative which could represent a high level class covering a number of quite diverse subject areas). The formula is simply intended to flag significant word matches and differentiate between these and insignificant matches where it would be inappropriate to assign the classmark or progress to further subclasses.

The formula is applied to the final score obtained from comparing every word in the document with every word in the class representative. If the result is greater than a set threshold, either the document is compared with subclasses of the current DDC class or the classmark of the current DDC class is assigned to the document (in the case of leaf nodes where there are no subclasses). The setting of a threshold is a common practice in information retrieval (van Rijsbergen 1981) The value of the threshold was set at 0.5 following some experimental adjustment (see appendix D). The formula is referred to as a significance test to differentiate it from a similarity measure. The formula denoting $SigTest(d_p,c_q)$ is an expression of the significance of the keyword matches between a document representation d_p and a class representative c_q. It reflects the relative importance between the frequency of words within a given document and their appearance in the class representative. The frequency of each word is weighted based on its appearance in the document as described in figure 12.

$$SigTest\ (d_p,\ c_q)$$

$$= 2\frac{\left[\left(W^{t}+10\right)\sum_{i=1}^{=lp} W^{t}_{ipq}\right]+\left[\left(W^{h1}+10\right)\sum_{i=1}^{=lp} W^{h1}_{ipq}\right]+\left[\left(W^{mn}+10\right)\sum_{i=1}^{=lp} W^{mn}_{ipq}\right]+\left[\left(W^{h2}+10\right)\sum_{i=1}^{=lp} W^{h2}_{ipq}\right]+\left[\left(W^{o}+10\right)\sum_{i=1}^{=lp} W^{o}_{ipq}\right]}{lp+lq}$$

where:

lp	=	length of the document d_p
lq	=	length of the class representative c_q
W^{t}	=	the weight given to document words appearing in the title
W^{h1}	=	the weight given to document words appearing in level 1 headings
W^{mn}	=	the weight given to document words appearing in meta tags
W^{h2}	=	the weight given to document words appearing in level 2 headings
W^{o}	=	the weight assigned to all other document words

$$W^{t}_{ipq}\begin{cases}1 & (word\ i \in class\ representative\ cq) \cap (word\ i \in tp)\\ 0 & otherwise\end{cases}$$

$$W^{h1}_{ipq}\begin{cases}1 & (word\ i \in class\ representative\ cq) \cap (word\ i \in h1p)\\ 0 & otherwise\end{cases}$$

$$W^{mn}_{ipq}\begin{cases}1 & (word\ i \in class\ representative\ cq) \cap (word\ i \in mnp)\\ 0 & otherwise\end{cases}$$

$$W^{h2}_{ipq}\begin{cases}1 & (word\ i \in class\ representative\ cq) \cap (word\ i \in h2p)\\ 0 & otherwise\end{cases}$$

$$W^{o}_{ipq}\begin{cases}1 & (word\ i \in class\ representative\ cq) \cap (word\ i \in op)\\ 0 & otherwise\end{cases}$$

tp = set of all words occurring in the title of document d_p.

$h1p$ = set of all words occurring in the first level heading of document d_p

mnp = set of all words occurring in meta tags of document d_p

$h2p$ = set of all words occurring in second level headings of document d_p

op = set of all other words occurring in document d_p

In each case the weight plus 10 is multiplied by the frequency. The 10 represents the weight acquired from the class representative where every word has a fixed weight of 10 (as set by experiments shown in appendix A). This assists the identification of significant word matches in broad vocabularies.

Example:

When the simple HTML file discussed in section 3.2.2.3 was run by the classifier as shown in figure 17, the result is that the document is classified under classmark "590 Animals". This is a comparatively simple example as this particular branch of the hierarchy only has one level below the top level superclass - 500 Natural Sciences. Obviously an industrial strength version of the classifier would define many subclasses of 590. The class 500 covers a diverse range of subject areas (mathematics, physics, chemistry, natural history, astronomy, earth sciences, palaeontology, biology, plants and animals) so it is a good example in that the top-level class has a diverse vocabulary.

The score resulting from the comparison between the document representative with the top-level class representative (500) is calculated as follows:

Title word frequencies:

Word	Hillside	Animal	Sanctuary
Frequency	1	1	1
Weight	10	10	10

First level heading word frequencies:

Word	Welcome	Hillside	Animal	Sanctuary
Frequency	1	1	1	1
Weight	10	10	10	10

Other words from ordinary paragraph frequencies:

Word	sanctuary	home	kinds	animals	dogs	cats	horses	rabbits	cows	sheep	pigs
Frequency	1	1	1	1	1	1	1	1	1	1	1
Weight	1	1	1	1	1	1	1	1	1	1	1

Note that in this instance there are no second level headings and no meta tags.

Firstly this document representative is compared with the class representative for 500 Natural Sciences. The intersection between these two classes is shown below:

Intersection word	animal	animals	dogs	cats	horses
Weight on class representative	10	10	10	10	10

The total number of words in the document is 17.

The total number of words in the class representative for 500 Natural Sciences is 93.

The significance test for the comparison between these two entities is therefore calculated as follows:

$$= 2\frac{10+10(1)+10+10(1)+1+10(4)}{17+93}$$

$$= 2\frac{20(1)+20(1)+11(4)}{110}$$

$$= 2\frac{20+20+44}{110}$$

$$= 2\frac{84}{110} = \frac{168}{110}$$

$$= 1.53$$

This is well above the set threshold of 0.5 so the document is then compared with each of the subclasses of 500 Natural Sciences. Clearly the only match it subsequently acquires is with the 590 Animals class where it in fact achieves a particularly significant score of 24.

3.2.4. Assigning Classmarks

Every time a significant match is found between a document and a DDC class representing a leaf node (with no subclasses), the corresponding DDC classmark is assigned to the vector of classmarks within the document object. When all significant paths through the DDC hierarchy have been followed, the classmarks that have been assigned to the document are organised into order according to score. The top classmarks (if there are more than one) can then be extracted.

3.2.5 The ACE Object

The ACE object is the application class that co-ordinates the whole classification process. It takes a number of command line arguments that enable the classification of a local or remote document:

```
java ace -url http://www.mysite.co.uk
```

will result in the classification of a remote document and:

```
java ace myhtmlfile.html
```

will result in the classification of a local file.

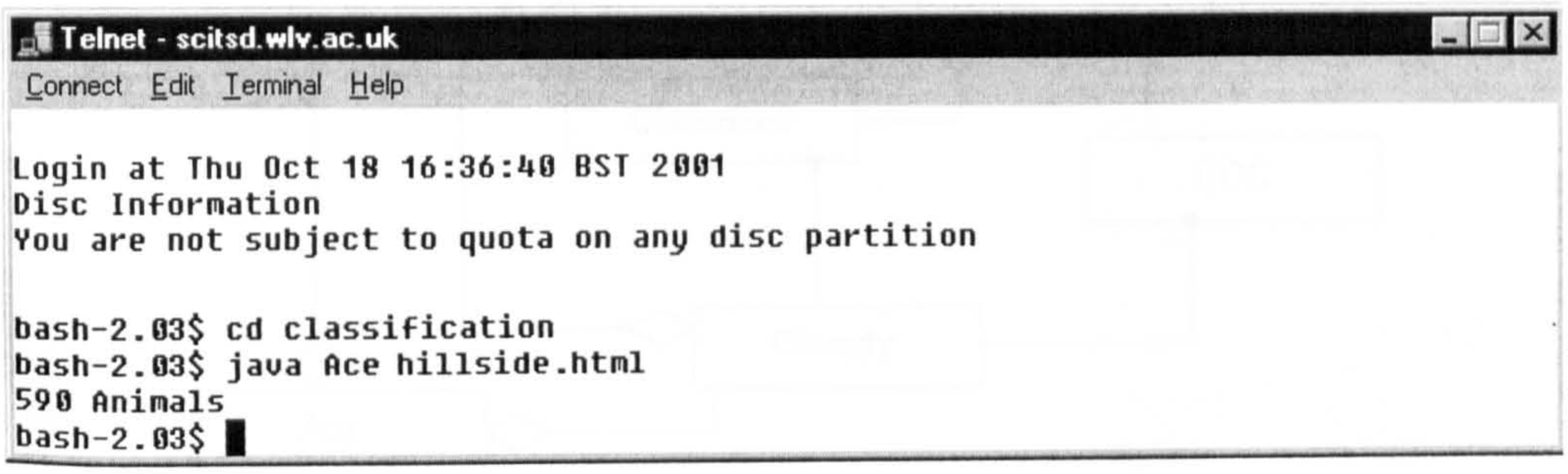


Figure 17. ACE classifying the example HTML page from 3.2.2.3

A connection is opened to the required document (local or remote) and an instance of the document object is instantiated. This automatically results in the construction of the document representative. The ACE object then creates an instance of the classify object, passing as a parameter the instance of the document object. This automatically triggers the classification process. Resulting classifications are acquired from the classify object and are output to the user.

3.2.6. Summary of the Classification procedure

When ACE is run, there are two main processes:

1. The generation of the Document Object.

   ACE is passed either a URL or the path to a local file as an argument. It uses which ever of these to create an instance of a Document object. The file is opened and parsed to create the object. This involves stripping out the HTML and storing each term in the Keywords vector with an assigned weight depending on where it was found and how often it appeared. ACE can also be sent a unique accession number as an argument (this is intended for use within WWLib-TNG, to enable unique identification of the document) which is also stored within the Document object.

2. The Document object is then passed as a parameter to the Classify object which co-ordinates the whole classification process, comparing the Document object with DDC objects down through the DDC class hierarchy as described above. This process results in classmark objects (from DDC objects representing significant leaf nodes) being assigned to the document.

The classmark objects found in the Document classmarks vector at the end of these two processes are then organised according to highest scores and the top scoring two (if there are more than one) are presented as appropriate classifications.

3.3 Detailed Design of New ACE

The classifier has an object-oriented design. DDC objects each inherit the same basic structure from a generic Dewey class and build their own list of Keyword objects forming the class representative, their own list of subclasses which are DDC objects in themselves and their own classmark object. Documents to be classified are represented as Document objects which comprise a list of weighted keywords, identical in structure to the DDC class representative. The result of the classification process

is that the document obtains a list of appropriate classmark objects each comprising a DDC classmark and accompanying label.

3.3.1 Objects and Classes

Figure 18 shows the main objects and the relationships between them in UML (Rambaugh et al. 1991) notation.

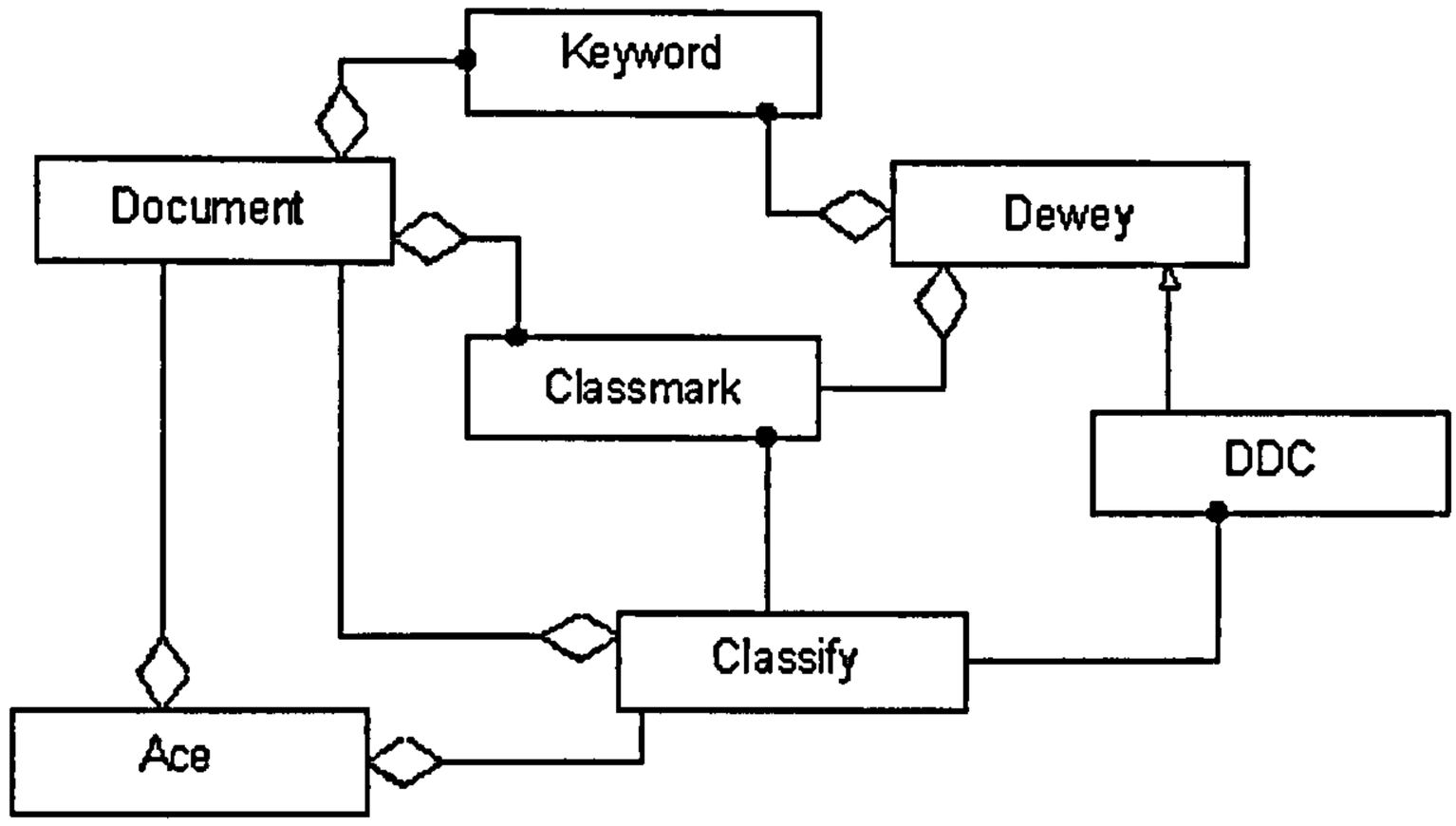


Figure 18. The main objects and the relationships between them.

The diagram shows that a document object is made up of many keywords and many classmarks and a Dewey object is made up of many keywords but only one classmark in each case. The classify object comprises a Document and is associated with many Dewey objects. The ace (Automatic Classification Engine) object is made up of a document object and a classify object. The following sections detail the classes required to implement these objects. The Java source code for each class can be found in Appendix E.

3.3.1.1 Classmark

Each classmark object stores and maintains a numerical code and a textual label representing a particular instance of a DDC classmark. It also stores an integer value - score - that represents the measure of similarity between the associated DDC object and a given document. Figure 19 shows the instance variables and methods associated with a classmark.

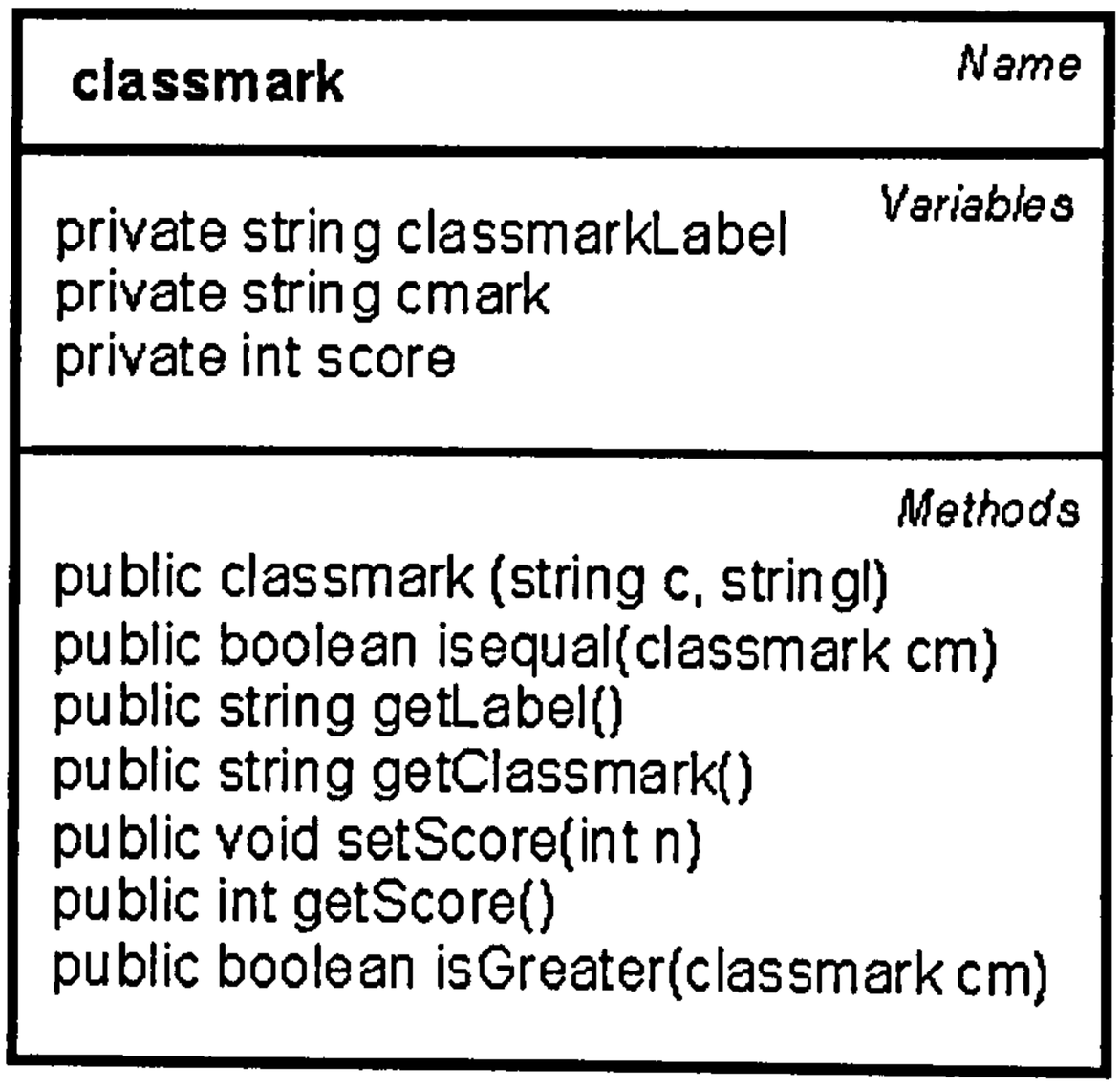


Figure 19. Classmark object

The constructor method takes as its parameters two strings, the first representing the numerical code which is subsequently stored in the cmark instance variable and the second the label which is then stored in the classmarkLabel instance variable. The method isequal returns true if the classmark object passed as a parameter is equal to the current one. The getLabel and getClassmark methods are there for retrieving information from the object. The methods setScore, getScore, and isGreater are used to store and compare measures of similarity between the associated DDC object and a given document, as a result of the classification process.

3.3.1.2 Keyword

Each keyword object stores and maintains a keyword string, a score (or weight - used by the document object to indicate significance according to frequency and/or position) associated with it and an integer representing the keyword's position (within a document). Figure 20 shows the instance variables and methods associated with a keyword.

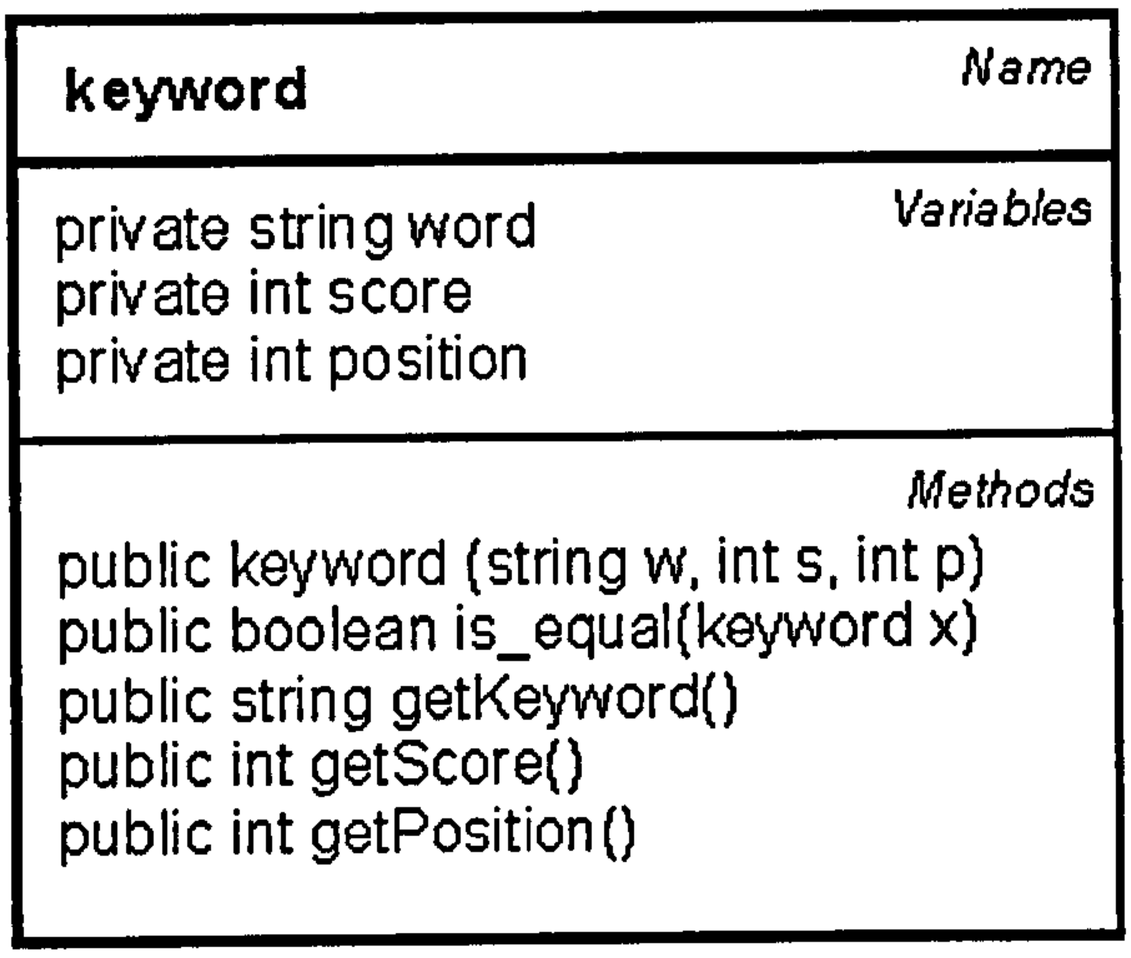


Figure 20. Keyword object

The constructor method takes as its parameters the word, score and position. Like the classmark object, the keyword object has an is_equal method for comparing two keywords and a getKeyword, getScore and getPosition method for retrieving information.

3.3.1.3 Dewey and DDC

Each DDC object inherits the abstract class Dewey which defines an object that stores and maintains a list of keywords, a list of subclasses and a classmark. Figure 21 shows the instance variables and methods associated with the abstract class Dewey.

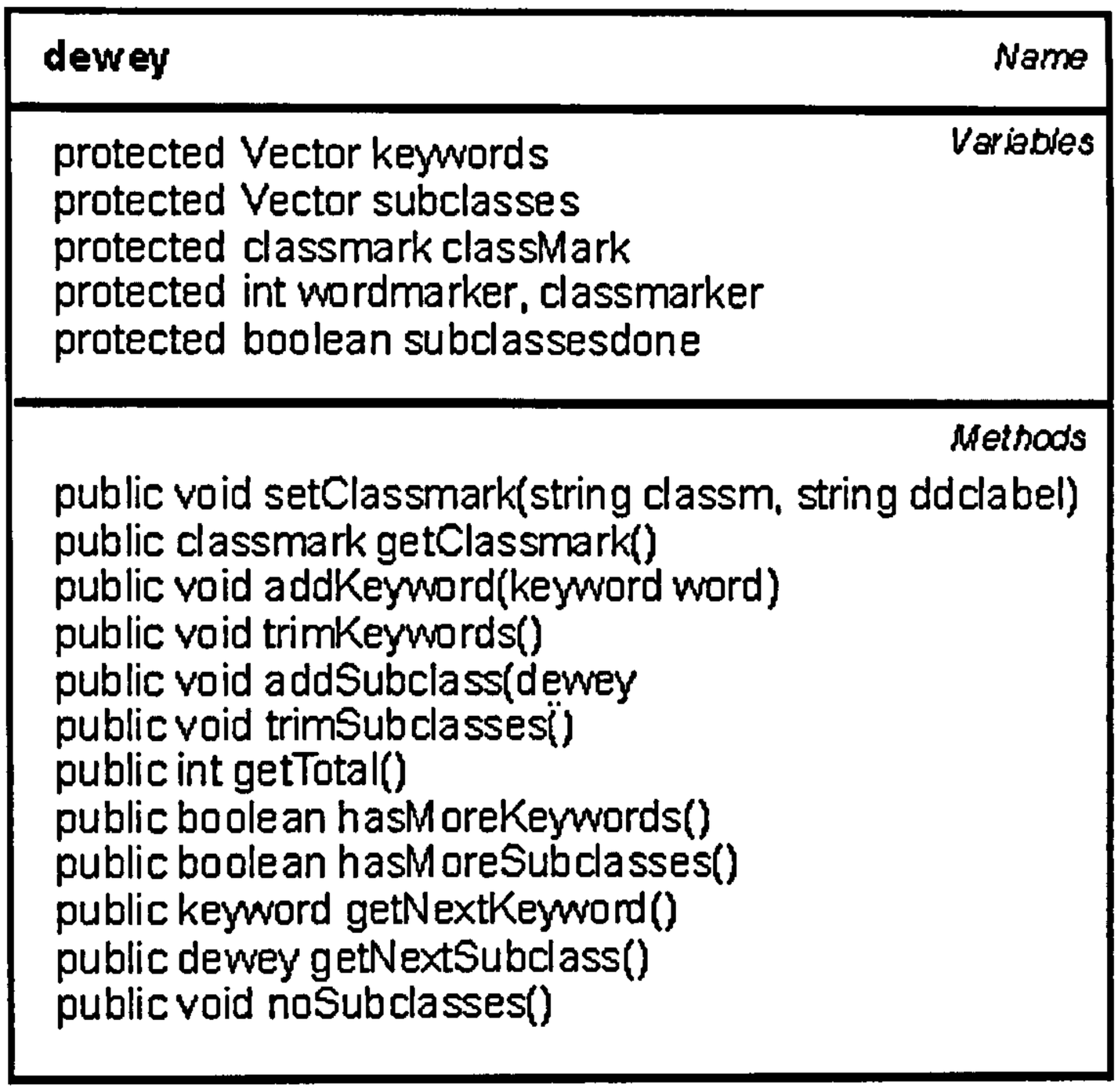


Figure 21. The abstract Dewey class

The methods enable each DDC object that inherits them to:

- specify its own classmark using setClassmark
- retrieve that classmark when required using getClassmark
- collate its own list of keywords (class representative) using addKeywords and trimKeywords
- collate its own list of subclasses using addSubclass and trimSubclasses
- retrieve the total number of keywords using getTotal
- retrieve keywords consecutively using getNextKeyword and hasMoreKeywords
- retrieve subclasses consecutively using getNextSubclass and hasMoreSubclasses
- Specify if there are no subclasses using noSubclasses

3.3.1.4 Document

Each document object stores and maintains a list of keywords representing the document and a list of classmarks, assigned to the document as a result of the classification process. Figure 22 shows the instance variables and methods associated with a document object.

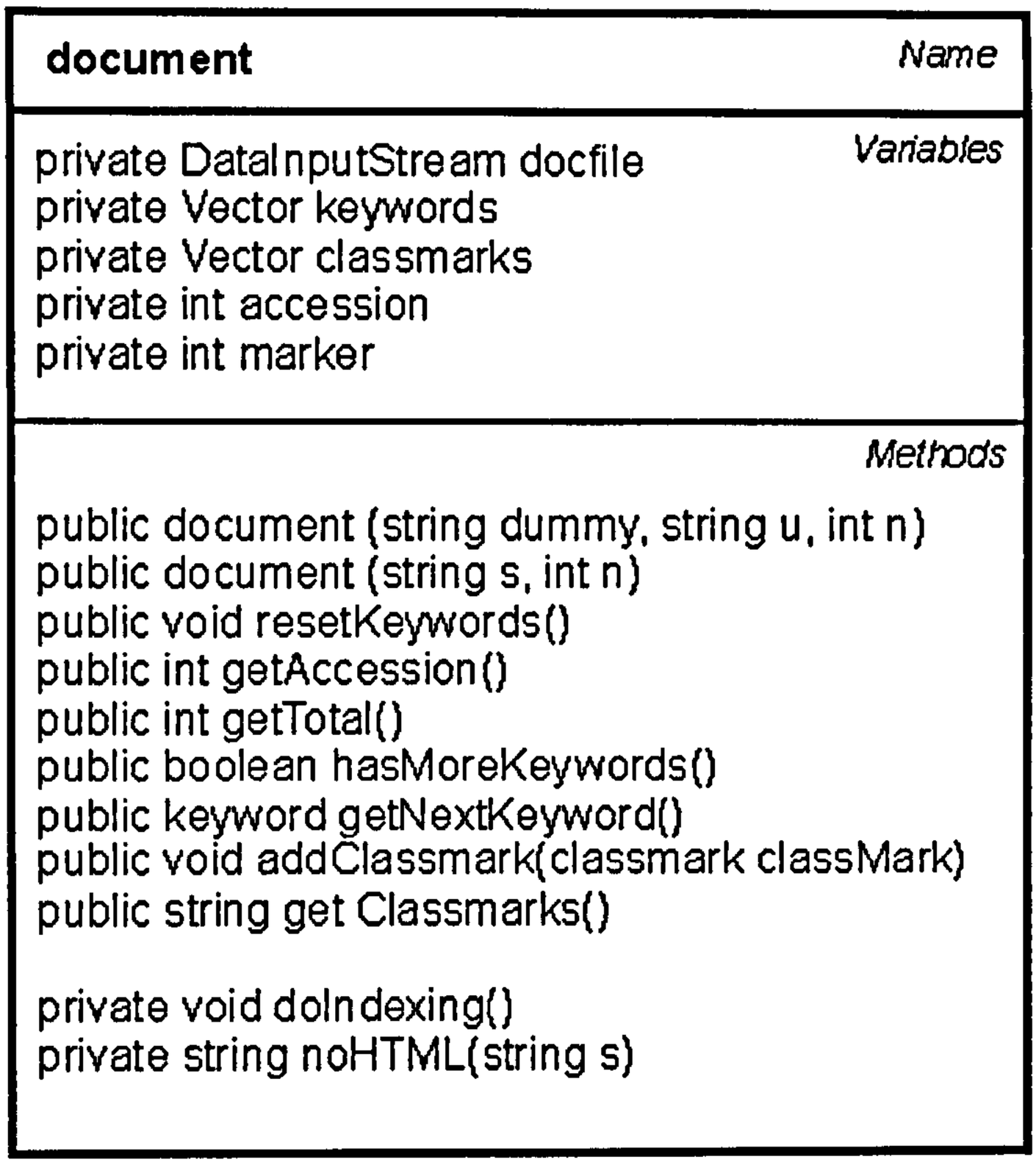


Figure 22. Document object

The document object has two constructor methods; one takes in three parameters - a dummy, a URL and a document accession number - and opens a DataInputStream to the remote URL; the other takes in two parameters -local filename and accession number - and opens a DataInputStream to a local file. In either case the methods open the document, extract all the words from it and build the keywords vector. This is done using the two private methods noHTML, which strips the HTML tags out of the document, and doIndexing which identifies all the remaining words and stores them as keyword objects with a weight, score and position in the keywords vector. Words occurring in titles and headings are given greater weight. All words are stored and those occurring frequently are automatically given more weight by appearing more often in the vector. The methods getNextKeyword, hasMoreKeywords and resetKeywords allow the contents of the keywords list to be retrieved and compared with DDC class representatives. The addClassmarks method is used by the classification process to assign classmarks that are found to be relevant to the document. The getClassmarks method retrieves the highest scoring classmark objects and outputs their numerical codes and labels as a string. getTotal retrieves the total number of keywords and getAccession retrieves the document accession number.

3.3.1.5 Classify

The classify object co-ordinates the whole classification process. It takes as its parameter a document object and compares its keywords with, initially, the class representatives of the ten top DDC classes (shown in figure 8). Each time a word in the document matches a word in the DDC class, the two associated weights are added to a total score. If the score is significant the classify object continues to recursively compare the document with that DDC object's subclasses. Figure 23 shows the attributes and methods associated with the classify object.

classify	*Name*
private document doc	*Variables*
public classify (document d) public string getClassmarks() private void proceed (dewey ddc) private boolean significant (int totalscore, int deweylength, int doclength) private int score (dewey ddc)	*Methods*

Figure 23. Classify object

The constructor method takes a document object as its parameter which is assigned to the doc instance variable. It then calls the private method proceed ten times passing a different DDC classification object (representing the top ten classes shown in figure 8) as its parameter each time. The proceed method takes the DDC object as its parameter, calls the private method score which compares the class representative of the object with the document and calculates a total score (using the private method score) based on matched keywords and associated weights. The score, together with the total length of the class representative and the total length of the document are then passed as parameters to the private method significant. The significant method calculates the above significance test and returns true if the resulting value is greater than the threshold. If significant returns true and the DDC object has (more) subclasses, the proceed method calls itself recursively on each of them. Otherwise the DDC object's classmark is added to the vector of classmarks in the document object. If significant returns false no further action is taken.

3.3.1.6 Automatic Classification Engine (ACE)

The ace object ties the whole system together. It accepts arguments from the command line and then creates a new document object using the given URL or filename, creates a new classify object using the new document object as its parameter and then outputs the resulting document classmarks. The ace object and its associated attributes and methods are shown in figure 24.

ace	*Name*
document doc classify classification	*Variables*
main (string args[])	*Methods*

Figure 24. The ace object

3.3.2 Networking ACE

In order to use and test the classifier independently, with the intention of presenting it to as wide an audience as possible, a client/server application was written. A user could type a URL into the client application, which sent the URL to the server - known as ThreadedAceServer - which then accessed the URL, classified it, and sent the classmarks back to the client to be presented to the user. Figure 25 shows the user interface of the client application.

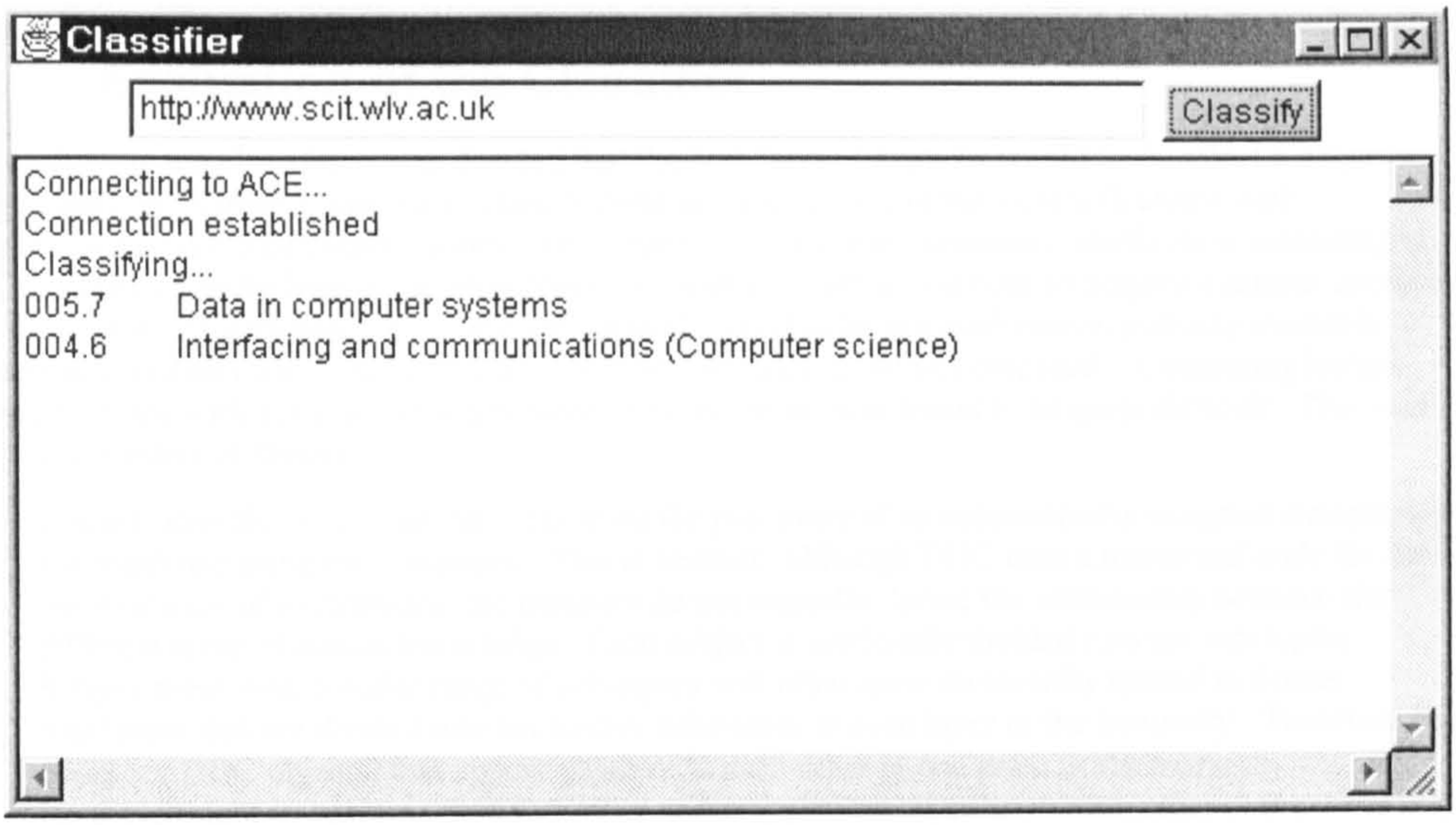


Figure 25. ACE client

The user entered the URL in the input box at the top, pressed the classify button and the client would then report what was going on and finally present the classifications in the text area below. The system worked quite well but required the user to have a copy of the Java Virtual Machine. An Applet version was written but this too would only work with certain browsers depending on how Applet security features were implemented. The classifier was later put on-line in a more reliable manner using a Java Servlet (see later section on automatic metadata generation). The classes required to implement the client/server application described here can be found in Appendix F.

3.4 Summary

The original classifier that was developed alongside WWLib-TOS used primarily just the DDC class labels to identify the classmarks assigned to a document. The original classifier (Old ACE) did not use the hierarchical structure of the DDC classification scheme to assist in the classification process. The new classifier, New ACE, uses class representatives comprising a full list of keywords and synonyms to model each node of the classification scheme within instances of the abstract class Dewey. These class representatives are compared with keywords representing the document to be classified using a recursive algorithm that filters the document through the classification hierarchy. Each time a significant match is found between the document and a DDC class representative, the document is compared with any subclasses of that class or the document is assigned the associated classmark in the case of leaf nodes. A special formula was devised that measures the significance of the keyword matches between a document and a class representative. This formula was derived from a similarity measure, the Dice Coefficient. Unlike Dice, the new formula enables terms within the document to be weighted according to frequency and according to appearance within particular HTML tags deemed to be useful in identifying significant terms. The new classifier is written in Java and has an object oriented design. Each class including all associated attributes and behaviours is described in this chapter.

An experiment was carried out where the performance of the new classifier was assessed by expert librarians. The next chapter discusses this experiment, providing an evaluation of the new classifier.

4. Evaluation of the Classifier

In evaluating the classifier it was decided that the best form of analysis would be to select a corpus of web pages, get human librarians to classify them and then compare their classifications with automatically assigned classifications. This approach was chosen because classification according to DDC has historically been a human process so it seemed most appropriate to acquire a human appraisal of the classifier's performance. There are currently no reliable, comprehensive, publicly available, automated systems with which the classifier's performance could be compared. Comparing human classifications with automatically generated ones, however, was found to be quite difficult. This was due to a number of factors:

- It is not possible to automatically calculate the proximity of an automatically assigned classmark to a manual one using the classmark. This is because, although DDC uses a numerical code for the identification of a classmark, the numbers do not logically define the relationship between the different areas of human knowledge. Each subject is artificially divided into ten sub-topics. Subject areas with a wider range of sub-topics will often quite awkwardly spread to deeper subclasses that are divided into ten further subclasses at each layer of the hierarchy. Therefore two nodes (or DDC objects) that appear adjacent to each other at one point in the hierarchy will often have a lot more in common than two other nodes appearing adjacent to each other at the same level of the hierarchy but under a different parent class. For example under the 500 Natural Sciences class subclasses 530 and 540 represent Physics and Chemistry respectively. Although all of the material classified under these classmarks will be roughly about science, the kind of material classified under 530 Physics will often be quite different in nature to the information stored under 540 Chemistry. At the same level of the hierarchy, but under the 200 Religion class, subclass 230 represents Christianity and Christian Theology and subclass 240 represents Christian Moral and Devotional Theology. It is likely that material classified under the two subclasses of Religion will have more in common than material classified under the two different subclasses of Natural Sciences - physics and chemistry. The Religion class is interesting in that subclasses 210, 220, 230, 240, 250, 260, 270 and 280 are all dedicated to Christianity with all other religions appearing in the subclasses of class 290 - "Comparative Religion and Other Religions". This is a clear indication of the cultural roots of DDC and how its subject divisions often have more to do with history than logic.
- The same material could be classified correctly under several different classmarks. Classification can be very subjective. Different people could classify the same material under different classmarks depending on their individual perspective and on their interpretation of the important aspects of the information. Choosing the most accurate classmark, out of several possibilities, can depend heavily on the individual differences of those doing the choosing. As the results discussed in this chapter show, even expert librarians often disagree on the appropriate classmark.
- Librarians are unaccustomed to classifying web material and often find the lack of information in shorter documents difficult to cope with. Some of the test data selected for this experiment was deemed to provide insufficient information to properly assess the automatic classifications. The lack of other clues such as a preface, information about the author, keywords and suggested classification assigned by the publisher also inhibits human classifiers.

This chapter discusses an experiment that was designed to evaluate the classifier (New ACE) while taking into account the above complexities where possible. Section 4.1 describes the design of the experiment. Section 4.2 describes how the test data was derived. Section 4.3 describes the instructions that were given to librarians from the University of Hull who assisted in the evaluation process by providing manual classifications and assessing the automatically generated ones. Section 4.4 presents the results from each librarian and the cumulative results from all three. Further analysis of patterns found within the results is also provided. Section 4.5 presents a summary of the data analysis.

4.1 Design of the Experiment

In order to evaluate the performance of the classifier under realistic conditions, an experiment with the following stages was devised:

1. Select a large corpus of about 20000 web pages, acquired by the WWLib-TNG spider component, and automatically classify them.

2. Randomly select a manageable subsection (about 200) of the pages that appear to have been successfully classified.
3. Ask some (about three) human classifiers to manually classify the same set of pages and to rate the automatically assigned classifications independently of each other.
4. Analyse the librarian ratings and look particularly at cases where they agreed or disagreed with the classifier and with each other.

The following sections will describe each of these stages in more detail, explaining exactly how each was implemented.

4.2 Selection of the Test Data

20000 web pages were acquired from the developer of the WWLib-TNG spider component. According to the developer, these pages were acquired by starting with a selection of "root" URLs from academic sites, zone transfers for org.uk and gov.uk primarily. These were then fed into the spider which fetched the pages, found new links and fed the new links back into the fetch procedure recursively for two or three iterations. The developer confirmed that he had no reason to suspect any significant subject area bias. The spider saves local copies of each page and generates a log file, which records an accession number, the URL of each page and the local file name where it has been saved.

For this evaluation experiment, a special Java application was developed that read the log file generated by the spider, opened a stream to the local copy of each page and classified it. The application generated its own results file which recorded: the accession number, the original URL acquired from the spider log file, the filename and path to the local copy, the classification classmarks and associated scores.

A second Java application was written which read the results file for the 20000 classifications and randomly selected a subset of 200 (source code for this application can be found in appendix G). This was done using the random number generator class that forms part of the Java API. A third Java application was written which copied each of the resulting 200 local copies into a location where it would be accessible by the librarians. The application also automatically generated an HTML "results" page which provided the librarians with, for each page, the accession number (1 - 200), a hyperlink to the local copy, a hyperlink to the remote version, a maximum of two automatically assigned classifications for each page and a series of empty columns. This HTML results page is shown in figure 26. The instructions accompanying this page are described in the following section. The librarians were asked to print this page so that they could use the empty columns to indicate a rating (by ticking under the associated column according to the instructions described in section 4.3) and write in a manual classification for each page.

4.3 Instructions to Librarians

Three librarians from the University of Hull, Scarborough Campus, Keith Donaldson Library, each of whom had experience in classification, agreed to assist in this evaluation experiment by providing manual classifications for each of the 200 randomly selected pages. Due to the difficulties in comparing manual classifications with automatically generated ones, discussed in the opening paragraphs of this chapter, the librarians were also asked to provide a rating of each automatic classification. It was hoped that this would enable classifications that were different to the manual classification but still not necessarily "wrong" to acquire some acknowledgement if the librarians saw fit.

The exact instructions that were provided to the librarians can be found in appendix H. They were instructed to access the "results page" shown in figure 26, and print it so that they had both the printed and on-line versions in front of them. They were then told to use the on-line version to follow hyperlinks and access the local copy of each page and use this to decide upon their manual classification which they were instructed to write on the paper version under the appropriate column. They were also instructed to provide a rating for each automatic classification by ticking under one of the rating columns labelled 1 - 4. Each page has a maximum of two classifications assigned by the classifier and each librarian was instructed to rate both of the possible two classifications independently. The first and second classmarks for each page are referred to collectively as classmark 1 and classmark 2 or first classmark and second classmark throughout this chapter. The rating scheme works as follows:

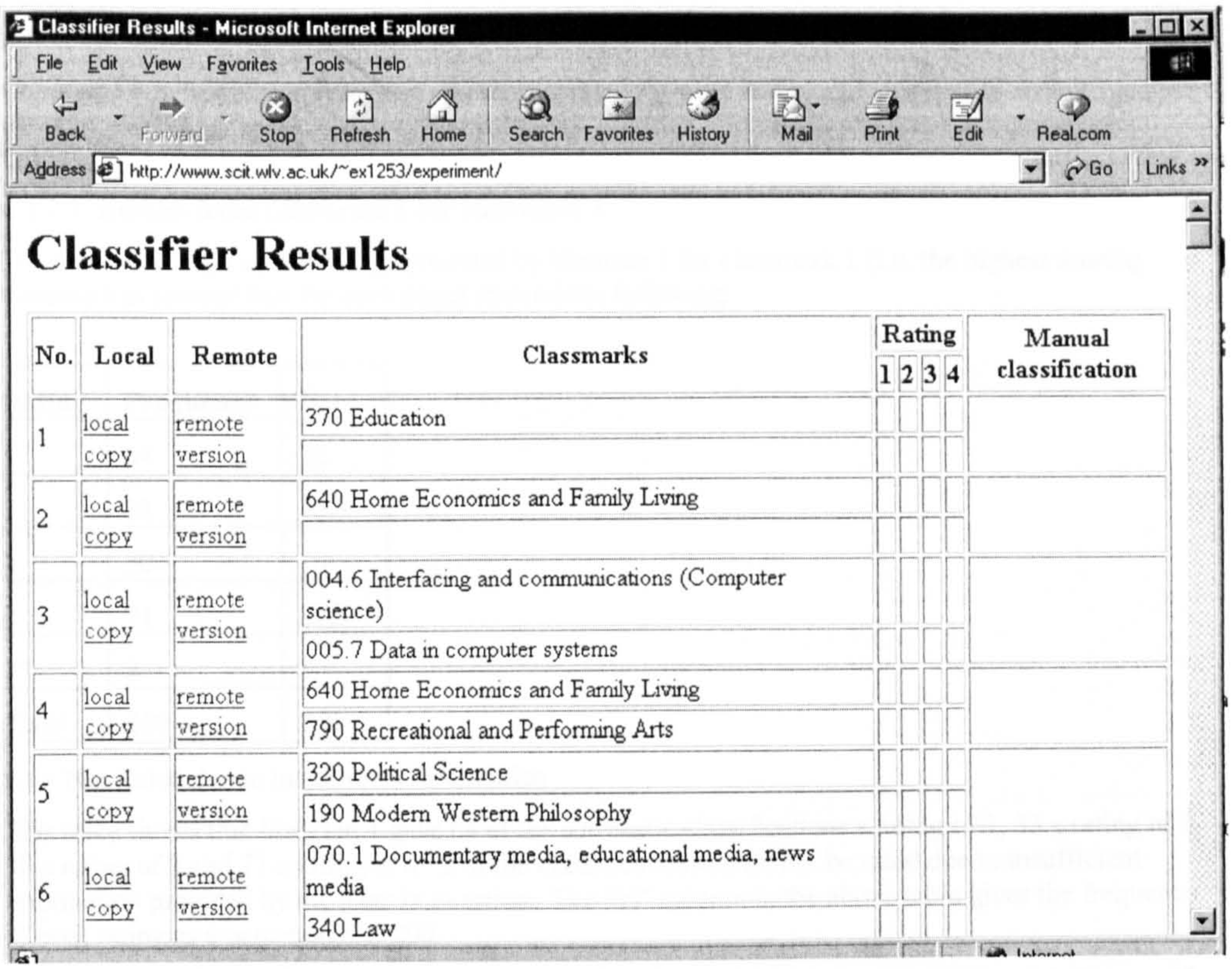


Figure 26. The first few rows of the "results page" used by librarians to access each of the pages used in the experiment and to assign ratings and manual classifications

1. This classification is completely inappropriate for this page.
2. Although not completely inappropriate, you would be surprised to see this page classified under this classmark in an on-line web directory.
3. Although not entirely accurate, you do not feel it would be misleading to see this page classified under this classmark in an on-line web directory.
4. This classification is accurate. This is where you would expect to see this page classified in an on-line web directory.

They were instructed only to follow the hyperlink to the remote version if the local copy was incomprehensible due to a lack of in-line images or other files supporting the interpretation of its content. It was stressed that the automatic classification was based on the content of the local copy and the remote version may have changed since it was stored by the spider so all ratings must be based on the content of the local version. They were shown how to "view source" where necessary in order to make more sense of local copies that appeared to show little evidence of content.

4.4 Results

Once the librarians had accessed each of the 200 pages and assigned a manual classification and a rating for each, they then returned their paper copies of the "results page" for analysis. Appendix I contains each of the results pages from each librarian and appendix J contains a spreadsheet where the data obtained from each of the three results sheets is integrated. The following sub-sections detail the findings obtained from the analysis of these results.

4.4.1 Results from each Librarian

Columns F - K from the spreadsheet shown in appendix J were sorted and analysed in order to acquire the number of 4,3,2 and 1 ratings assigned by each librarian. The results of this exercise were as follows:

4.4.1.1. Results from Librarian 1 for classmark 1

The result of analysing the ratings provided by librarian 1 for classmark 1 (i.e. the highest scoring classmark presented first for each page) showed the following:

Rating	Frequency	%
1	64	32
2	33	16.5
3	30	15
4	71	35.5
5*	2	1
Total	*200*	*100*

*5 = Not Rated due to insufficient information

This table shows that librarian 1 gave 64 of the automatic classifications a rating of 1, 33 a rating of 2, 30 a rating of 3 and 71 a rating of 4. 2 of the classifications could not be rated due to insufficient information provided by the page in question. The "%" column in the above table gives the frequency of each rating as a percentage of 200.

This shows 50.5% were considered to be good classifications, acquiring ratings of 3 and 4. Figures 27 and 28 show graphical bar and pie chart representations of the data from this table.

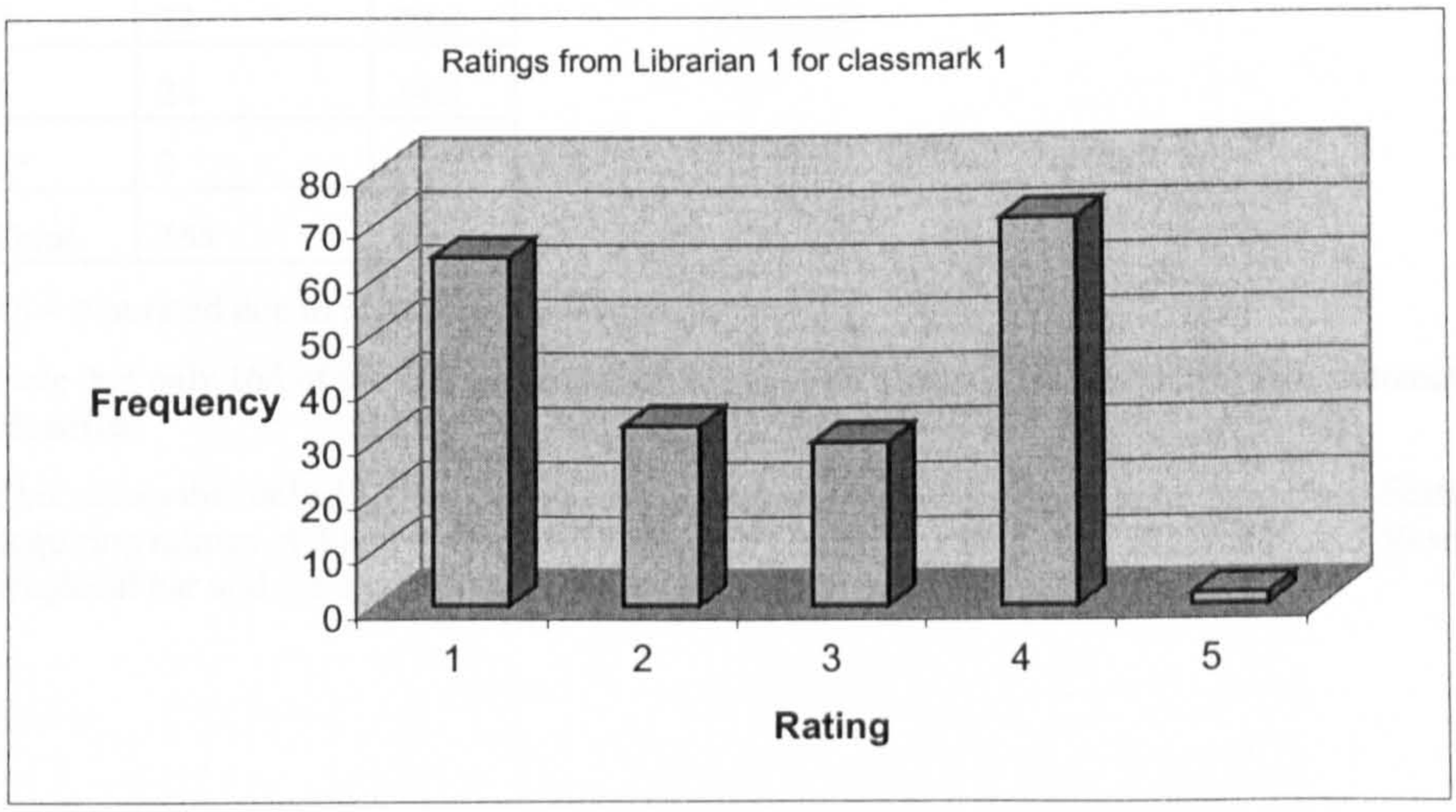


Figure 27. Bar Chart showing ratings from librarian1 for classmark 1

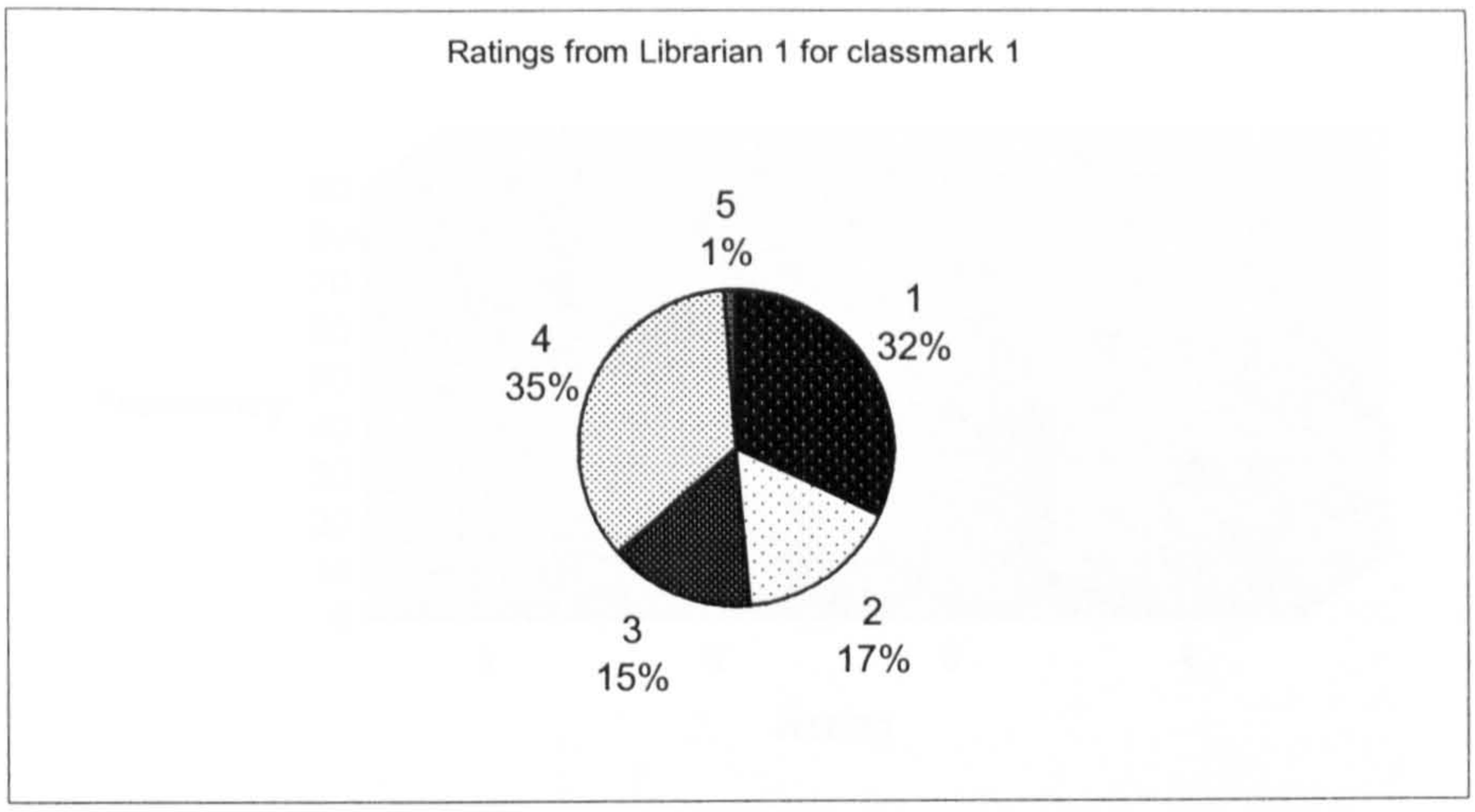


Figure 28. Pie chart showing ratings from librarian 1 for classmark 1

4.4.1.2. Results from Librarian 1 for classmark 2

The result of analysing the ratings provided by librarian 1 for classmark 2 (i.e. the second highest scoring classmark presented second for each page) showed the following:

Rating	Frequency	%
1	86	52.1
2	21	12.7
3	34	20.6
4	24	14.5
5*	0	0
Total	*165*	*100*

*5 = Not rated due to insufficient information

Note that only 165 of the 200 pages classified acquired a second classmark from the automatic classifier.

This shows that only 35.1% of the second classmarks were considered to be good classifications, acquiring ratings of 3 and 4. Over 50% acquired a negative rating of 1. Figures 29 and 30 show graphical bar and pie chart representations of the data from this table.

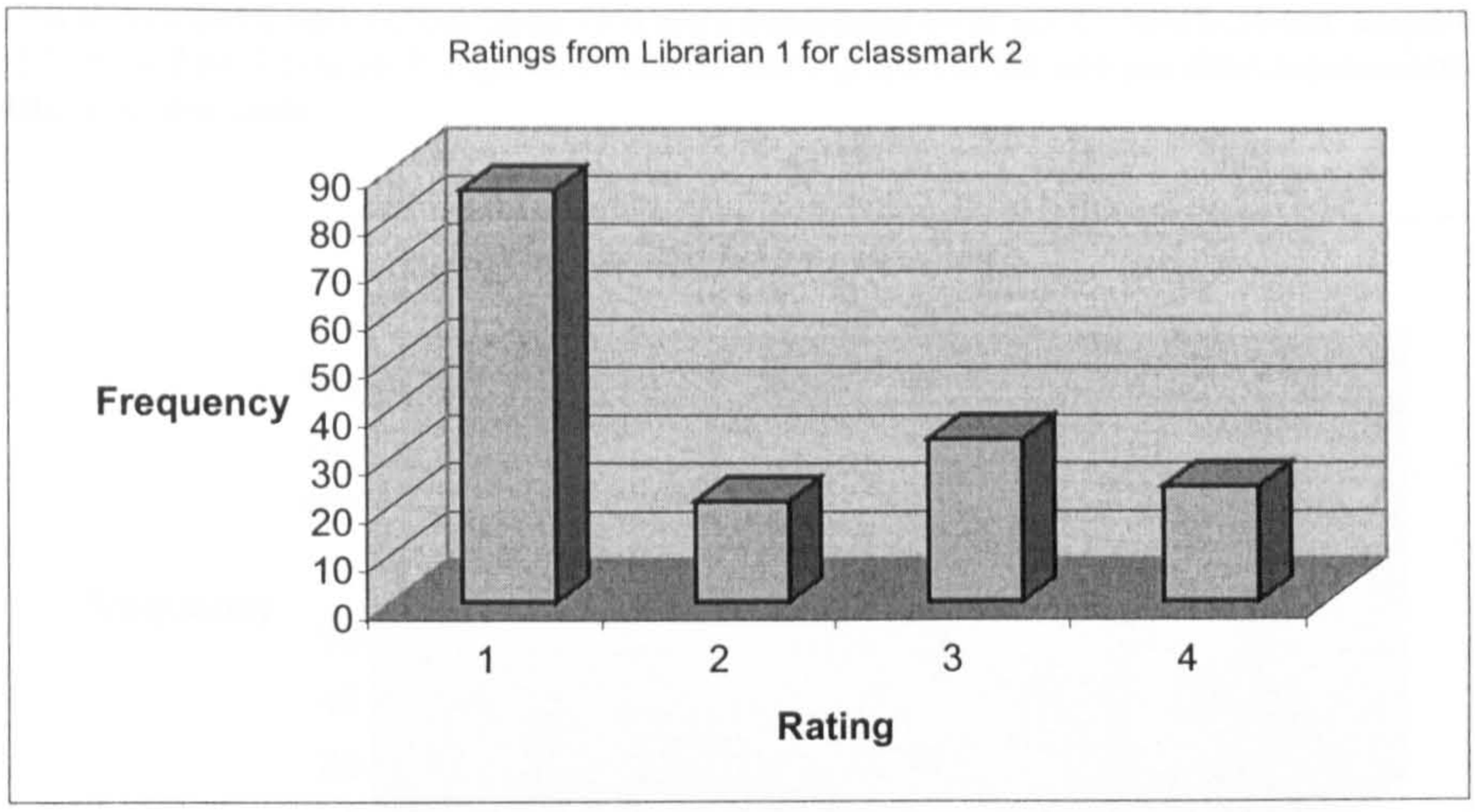


Figure 29. Bar chart showing ratings from librarian 1 for classmark 2

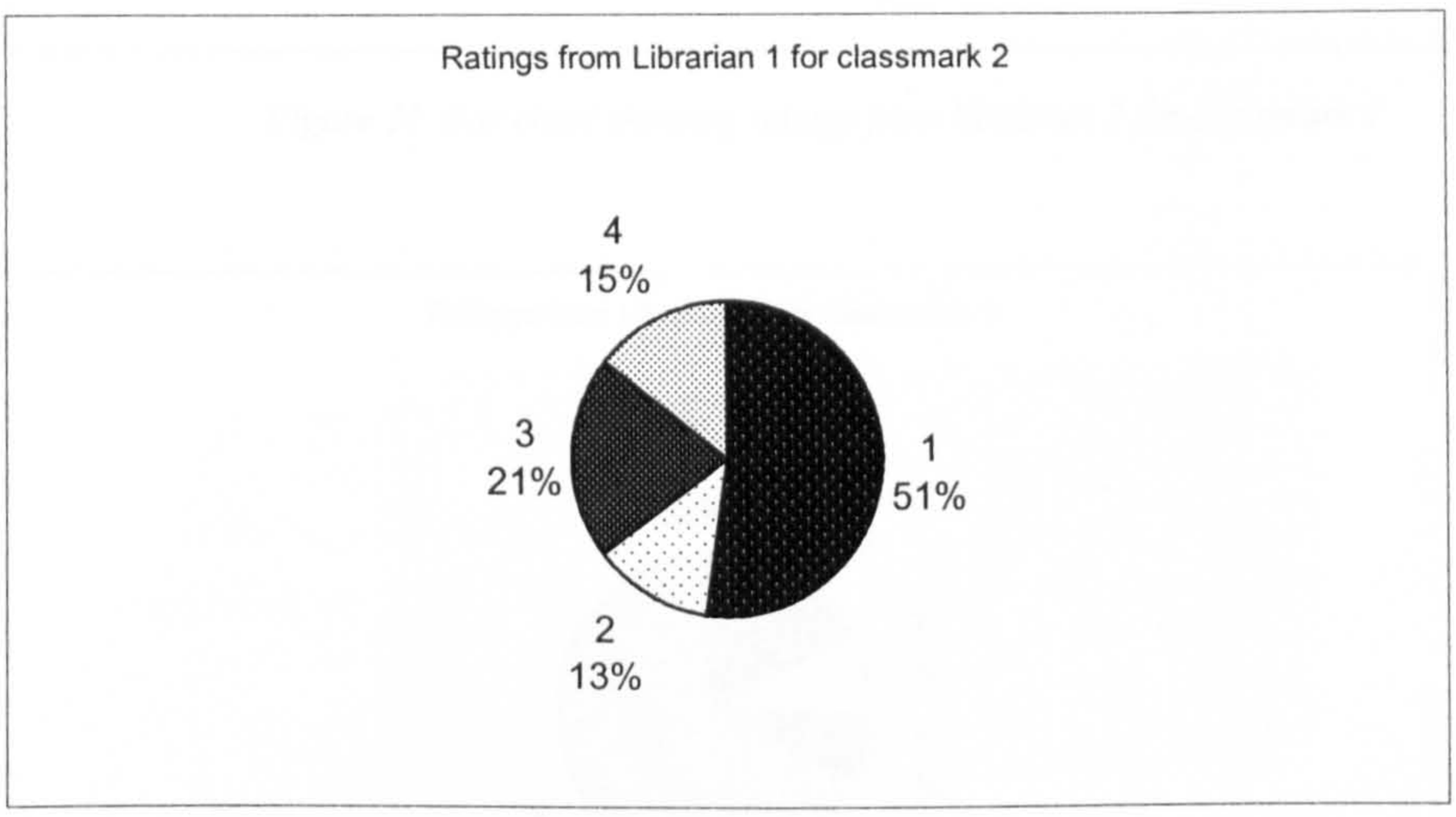


Figure 30. Pie chart showing ratings from librarian 1 for classmark 2

4.4.1.3. Results from Librarian 2 for classmark 1

The result of analysing the ratings provided by librarian 2 for classmark 1 showed the following:

Rating	Frequency	%
1	34	17
2	17	8.5
3	14	7
4	132	66
5*	3	1.5
Total	*200*	*100*

*5 = Not rated due to insufficient information

This shows that a very encouraging 73% were considered to be good classifications, acquiring ratings of 3 and 4 from librarian 2. Figures 31 and 32 show graphical bar and pie chart representations of the data from this table.

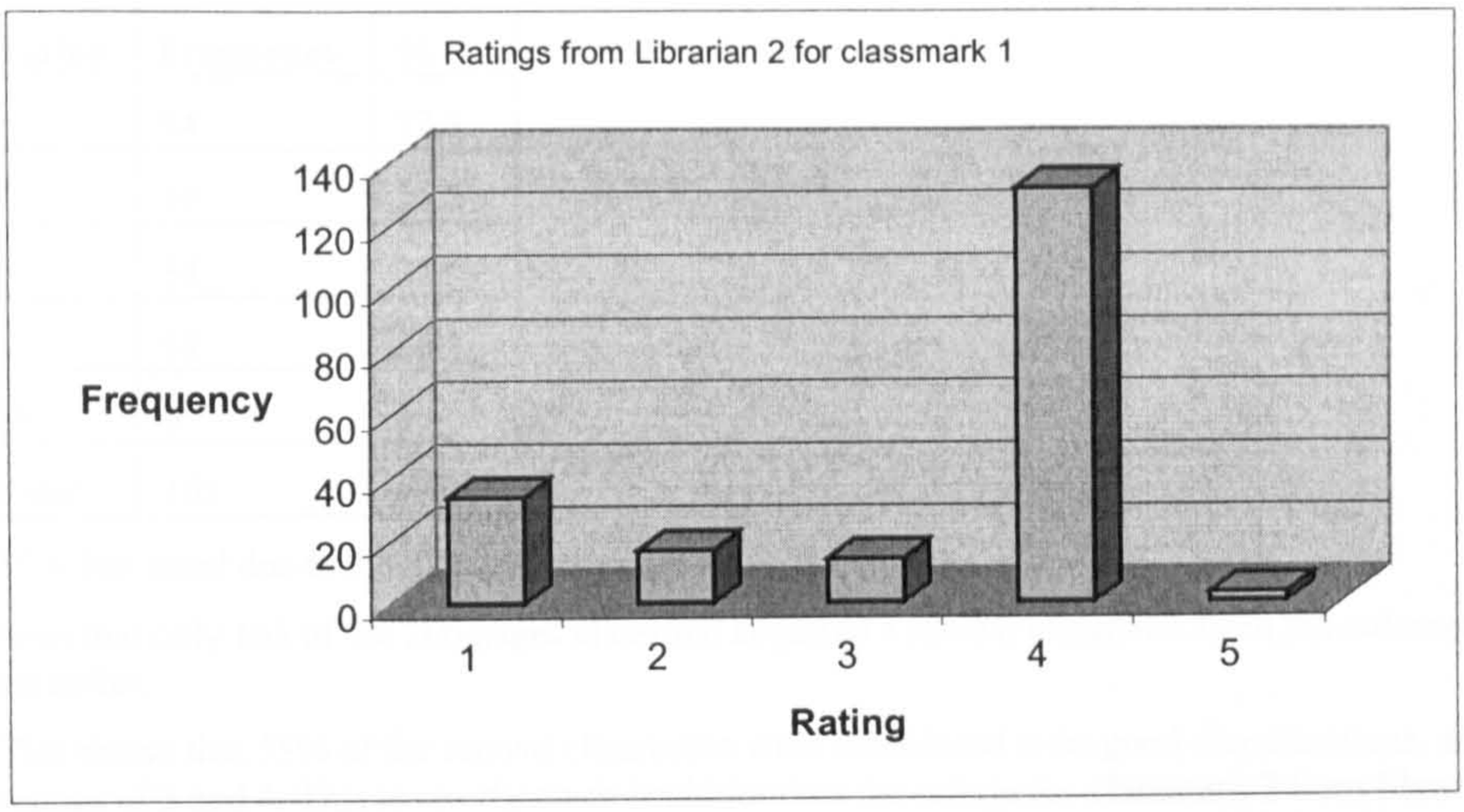


Figure 31. Bar chart showing ratings from librarian 2 for classmark 1

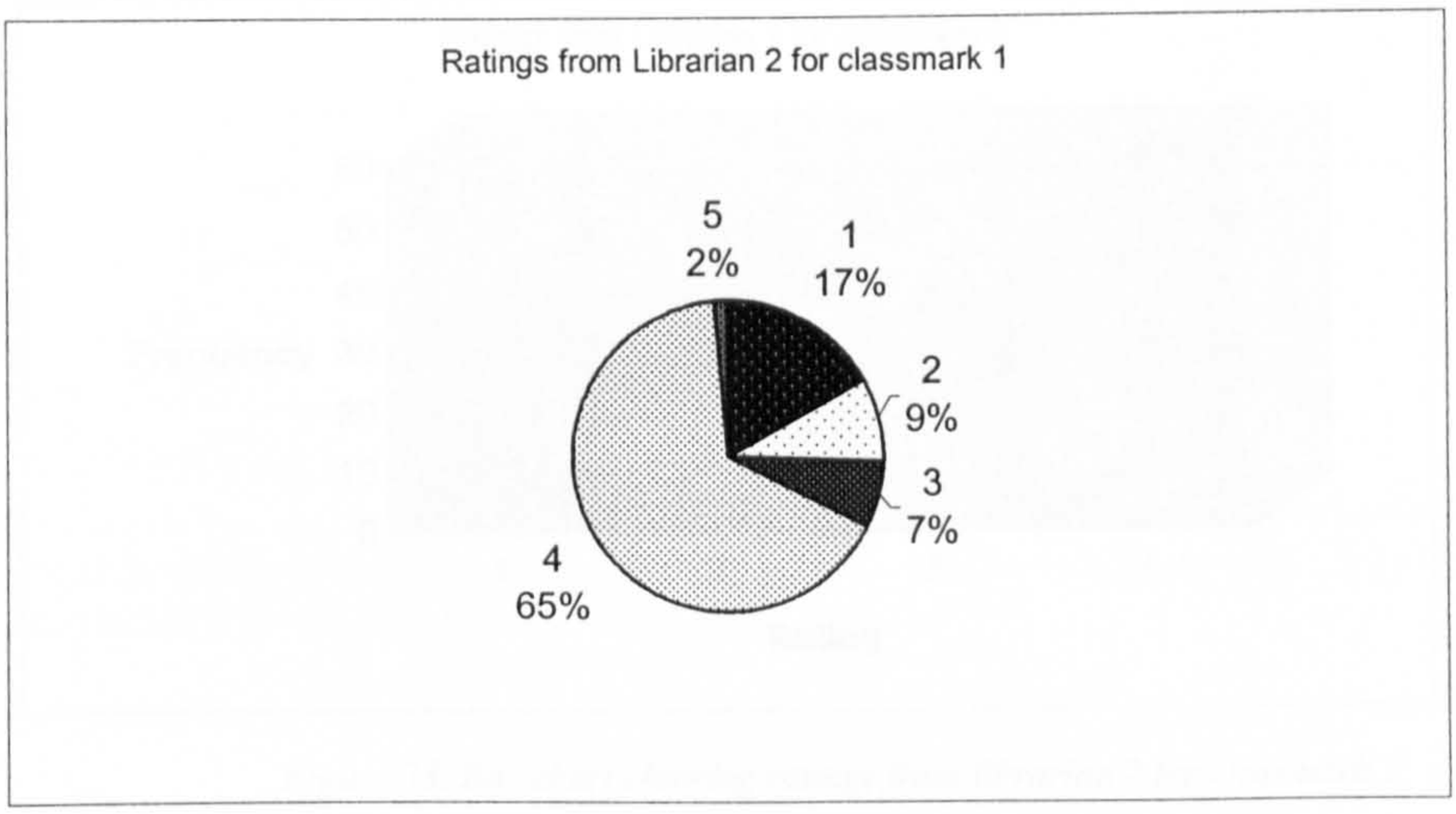


Figure 32. Pie chart showing ratings from librarian 2 for classmark 1

4.4.1.4. Results from Librarian 2 for classmark 2

The result of analysing the ratings provided by librarian 2 for classmark 2 showed the following:

Rating	Frequency	%
1	54	32.7
2	19	11.5
3	34	20.6
4	58	35.1
5*	0	0
Total	*165*	*100*

*5 = Not rated due to insufficient information

Note that only 165 of the 200 pages classified acquired a second classmark from the automatic classifier.

This shows that 55% of the second classmarks were considered to be good classifications, acquiring ratings of 3 and 4. This is clearly much healthier than the ratings for classmark 2 from librarian 1 where over 50% acquired a negative rating. Figures 33 and 34 show graphical bar and pie chart representations of the data from this table.

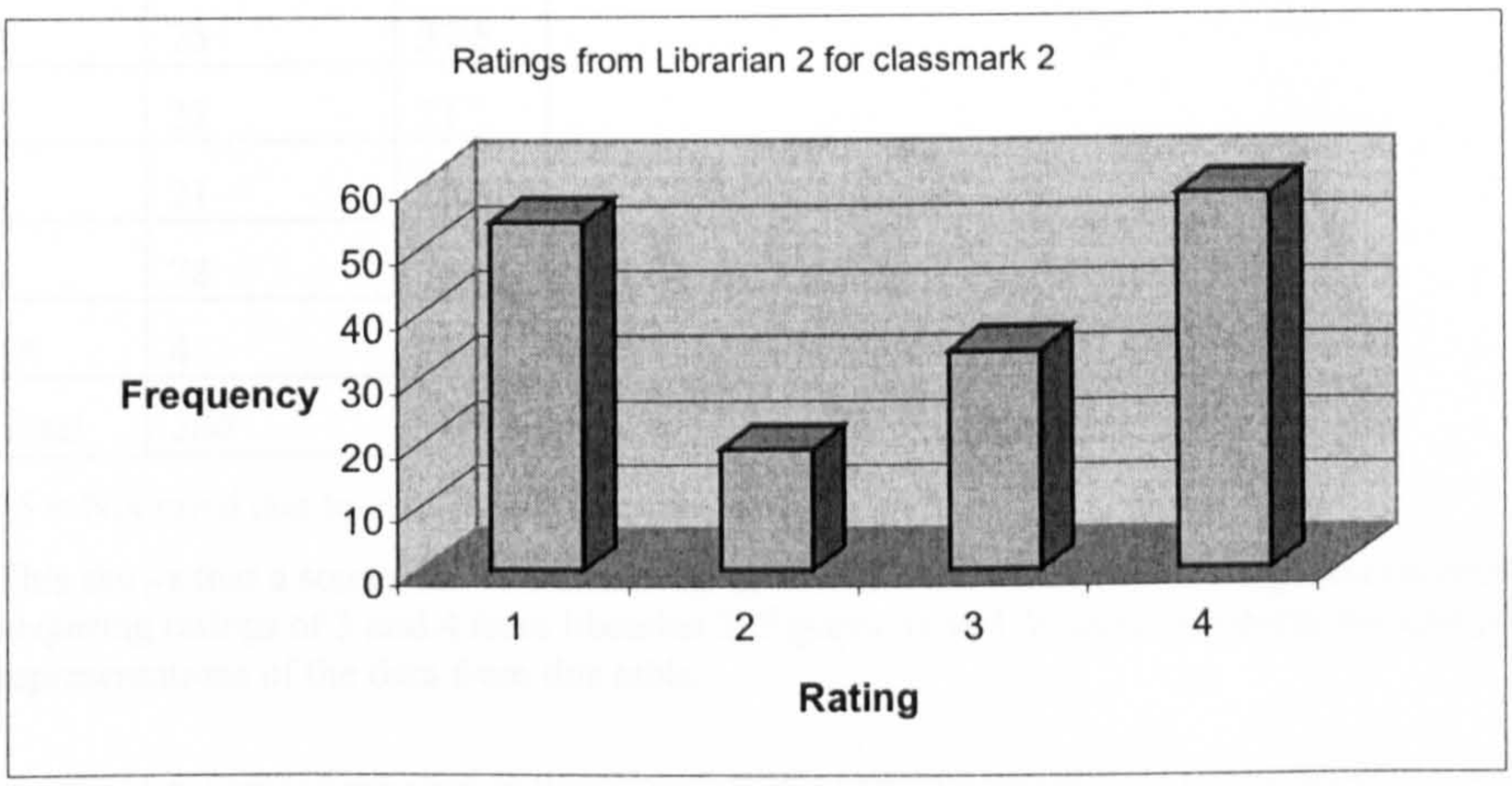


Figure 33. Bar chart showing ratings from librarian 2 for classmark 2

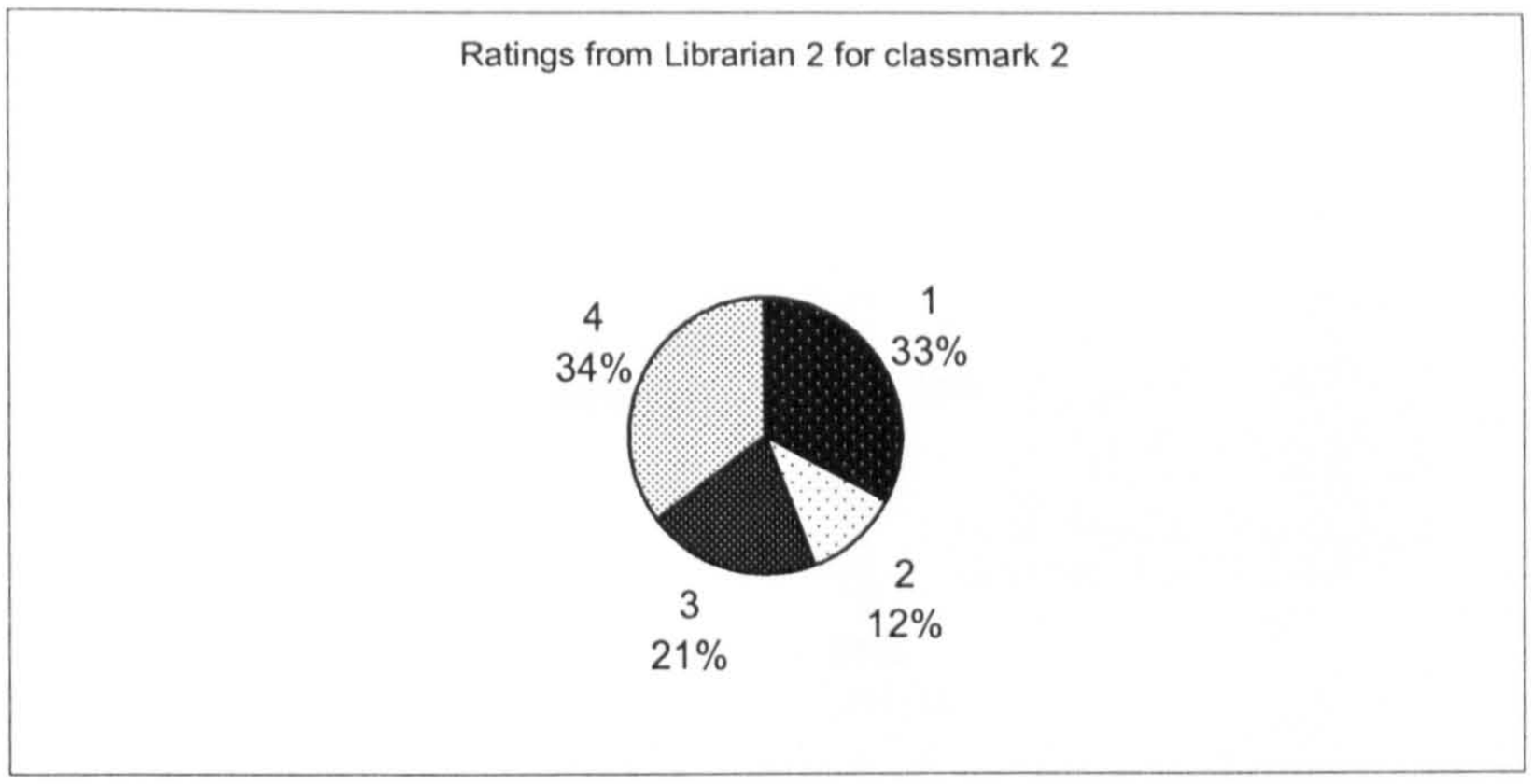


Figure 34. Pie chat showing ratings from librarian 2 for classmark 2

4.4.1.5. Results from Librarian 3 for classmark 1

The result of analysing the ratings provided by librarian 3 for classmark 1 showed the following:

Rating	Frequency	%
1	75	37.5
2	22	11
3	21	10.5
4	78	39
5*	4	2
Total	*200*	*100*

*5 = Not rated due to insufficient information

This shows that a somewhat less encouraging 49.5% were considered to be good classifications, acquiring ratings of 3 and 4 from librarian 3. Figures 35 and 36 show graphical bar and pie chart representations of the data from this table.

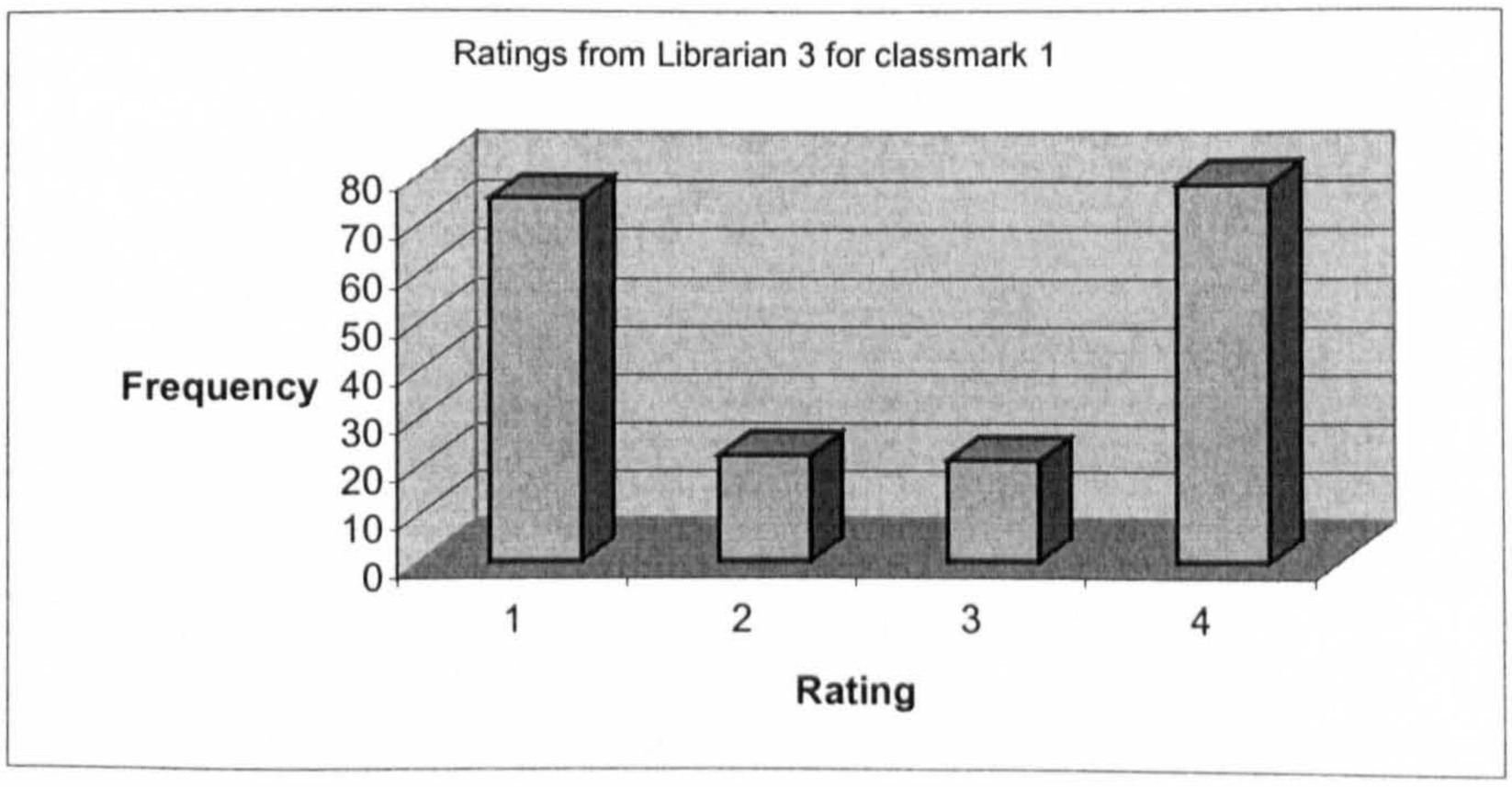


Figure 35. Bar chart showing ratings from librarian 3 for classmark 1

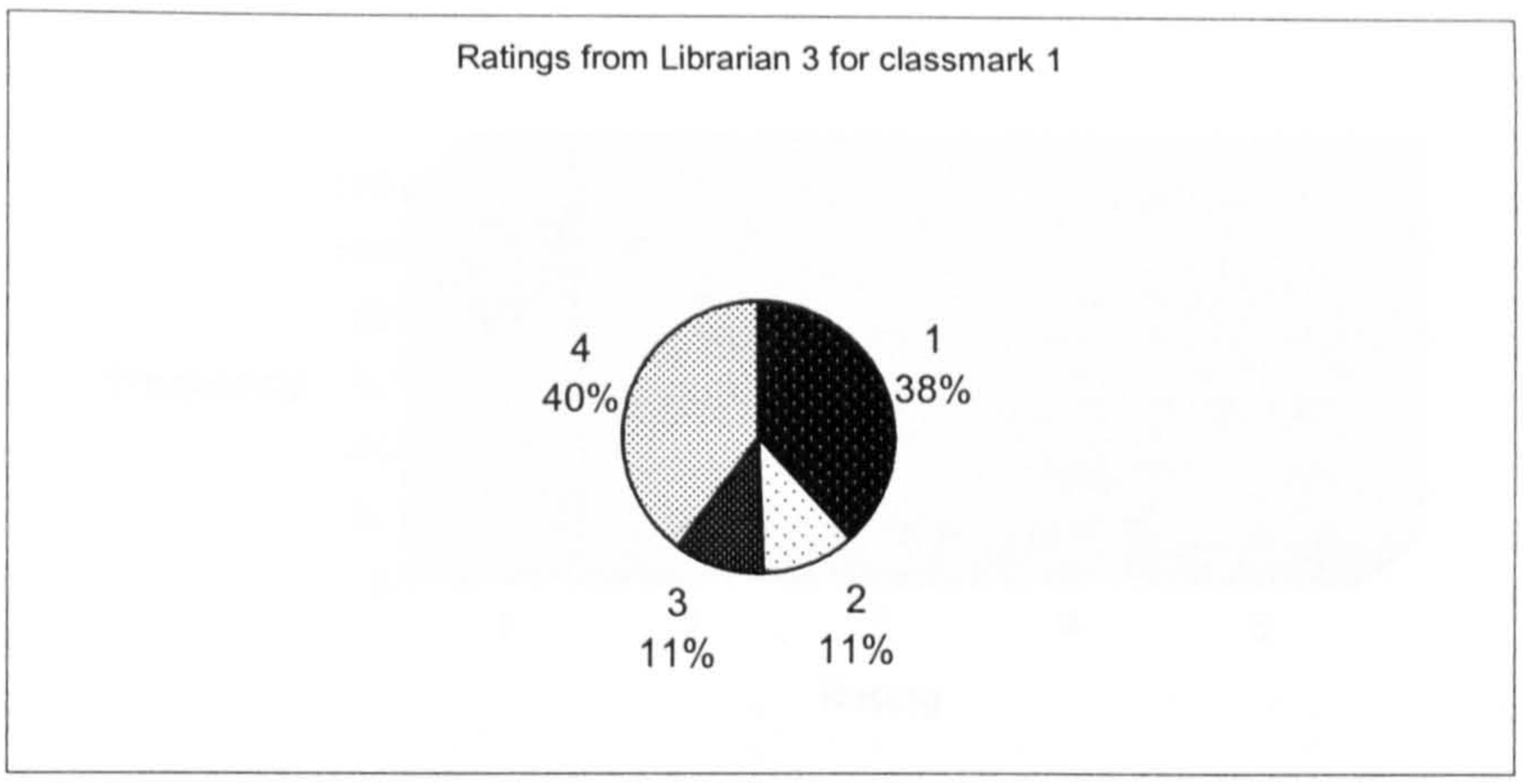


Figure 36. Pie chart showing ratings from librarian 3 for classmark 1

4.4.1.6. Results from Librarian 3 for classmark 2

The result of analysing the ratings provided by librarian 3 for classmark 2 showed the following:

Rating	Frequency	%
1	104	63
2	18	10.9
3	13	7.8
4	26	15.7
5*	4	2.4
Total	*165*	*100*

*5 = Not rated due to insufficient information

This shows that just 23.5% of the second classmarks were considered to be good classifications by librarian 3, acquiring ratings of 3 and 4. Figures 37 and 38 show graphical bar and pie chart representations of the data from this table.

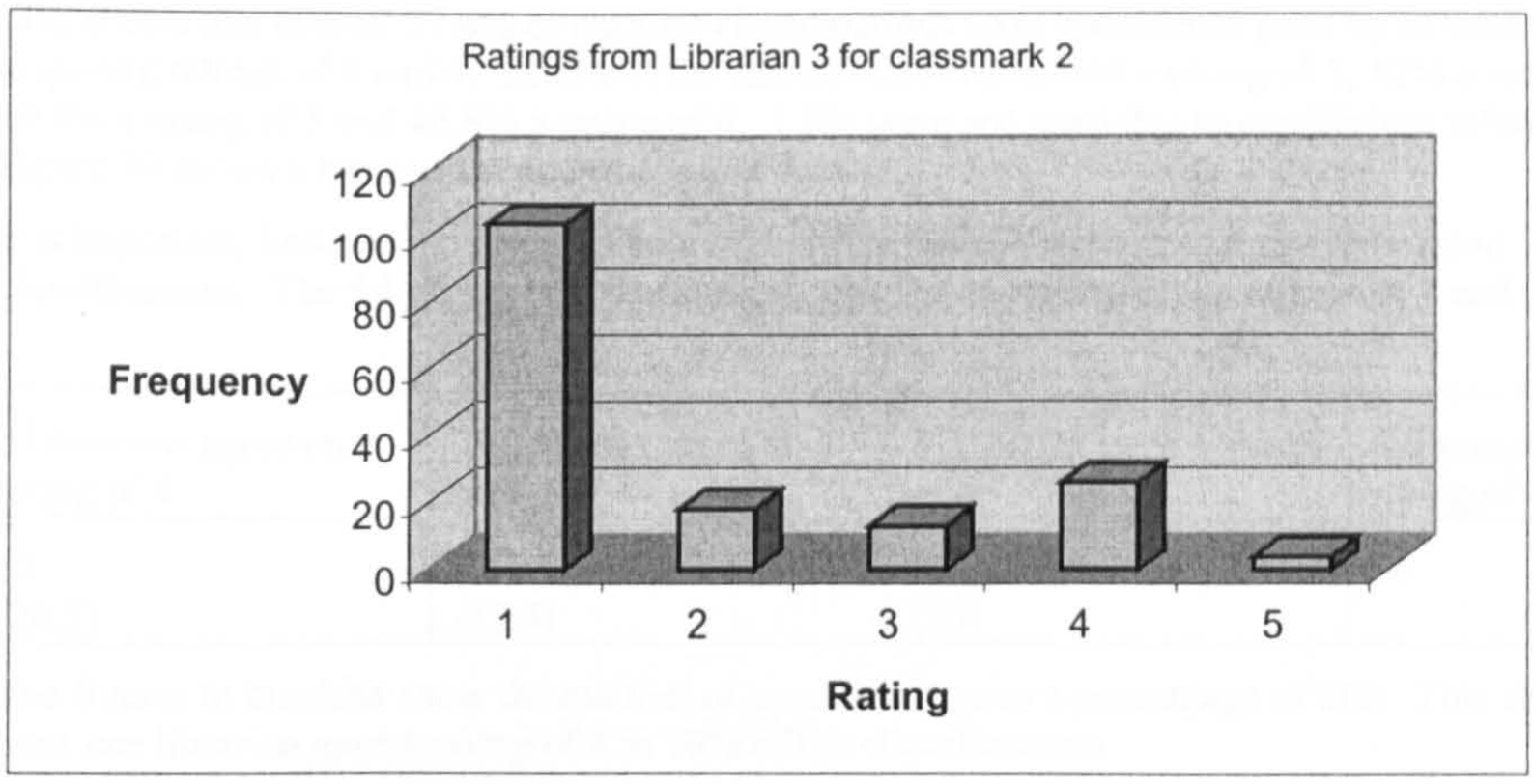


Figure 37. Bar chart showing ratings from librarian 3 for classmark 2

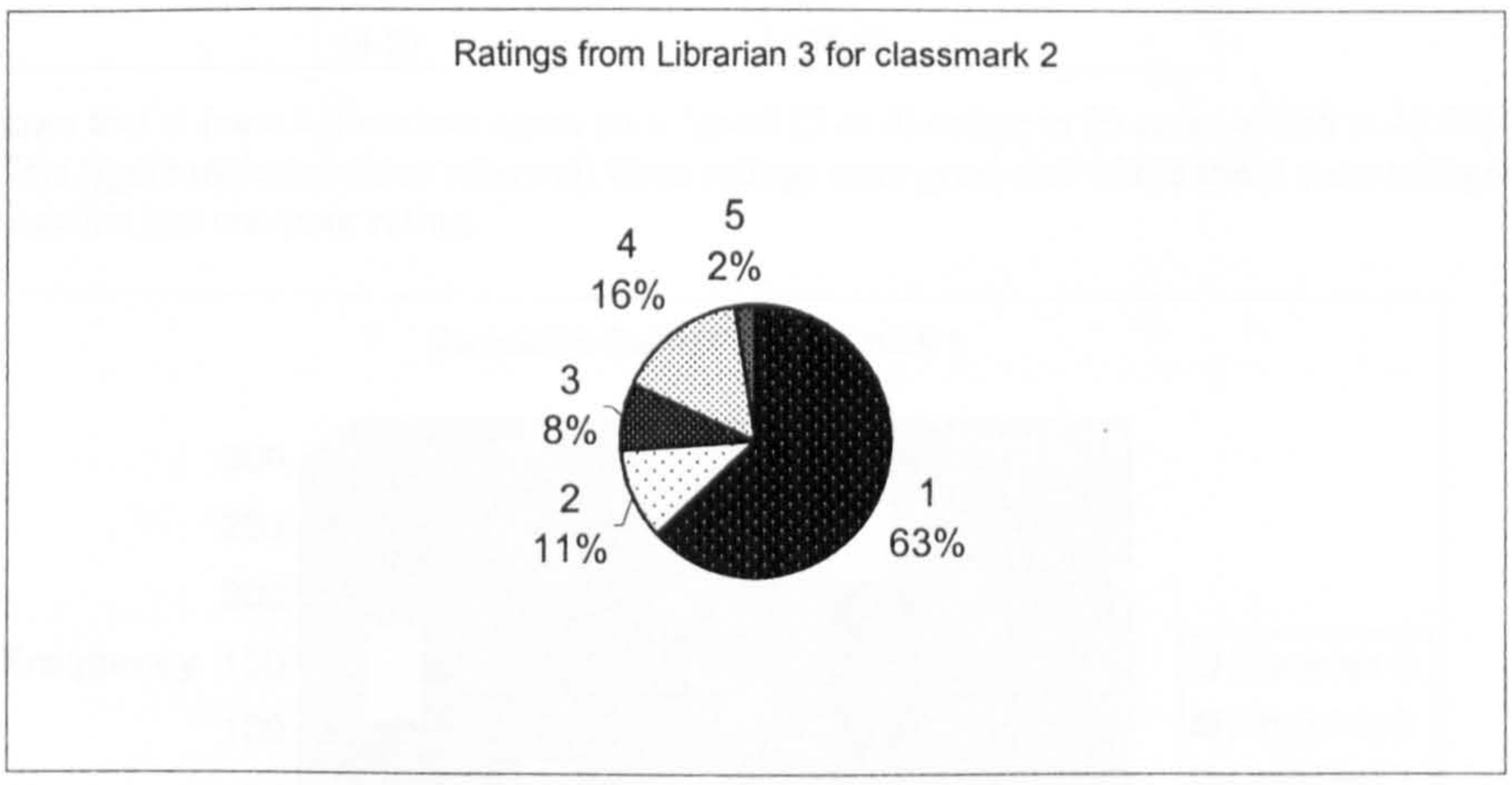


Figure 38. Pie chart showing ratings from librarian 3 for classmark 2

4.4.2 Cumulative Results

The result of combining the individual results from each librarian as shown in the previous section gives the following:

4.4.2.1 Cumulative Results for classmark 1

Rating	Librarian 1	Librarian 2	Librarian 3	Total	%
1	64	34	75	173	28.8
2	33	17	22	72	12
3	30	14	21	65	10.8
4	71	132	78	281	46.8
5*	2	3	4	9	1.5
Total	*200*	*200*	*200*	*600*	*100*

* 5 = Not rated due to insufficient information.

This shows that overall 57.6% of the first classifications were considered good by at least one librarian, acquiring ratings of 3 and 4. 28.8% of the classifications acquired a rating of 1, 12% a rating of 2, 10.8% a rating of 3 and 46.8% a rating of 4. 1.5% were not rated due to insufficient information. Figure 39 shows a bar chart representing this data.

It is important, however, to consider how often the librarians were in agreement on good classifications. The following tables analyse the degree of agreement on ratings of 4 and 3:

3 librarians agree on a rating of 4	2 librarians agree on a rating of 4	1 librarian gave a rating of 4	0 librarians gave a rating of 4
53 (26.5)	35 (17.5)	52 (26)	60

The figures in brackets show the number of agreed ratings as a percentage of 200. This shows that at least one librarian gave a rating of 4 in 70% of the classifications.

3 librarians agree on a rating of 3	2 librarians agree on a rating of 3	1 librarian gave a rating of 3
2 (1)	9 (4.5)	41 (20.5)

This shows that at least 2 librarians agree on a 'good' (3 or 4) rating in 99 cases which is 49.5% of the time. This figure includes cases where all three ratings were good and where there were two good ratings against just one poor rating.

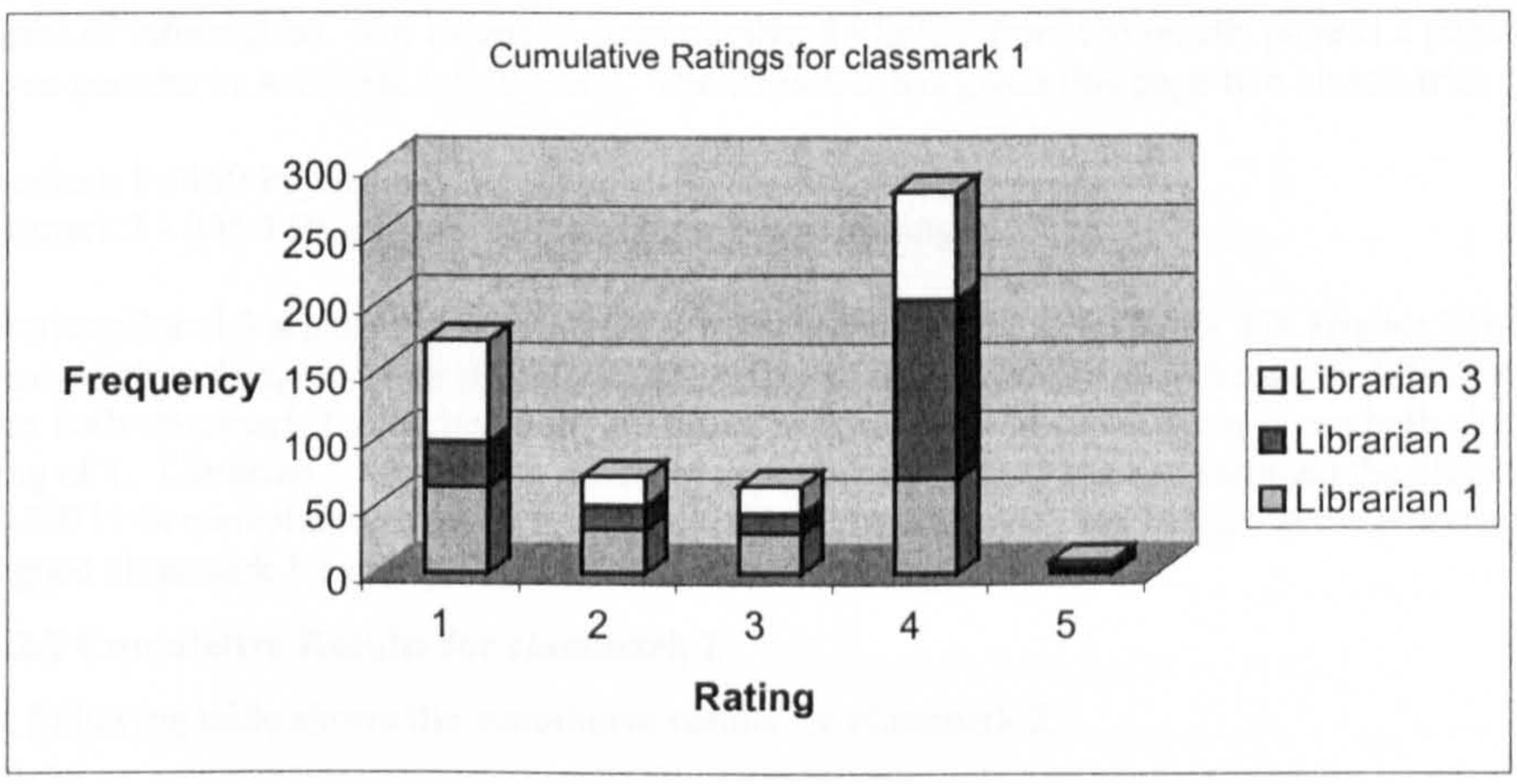


Figure 39. Bar chart showing cumulative ratings for classmark 1

Figure 40 shows a doughnut chart of the same data. The ratings for librarian 1 are represented by the inner ring, librarian 2 by the middle ring and librarian 3 by the outer ring.

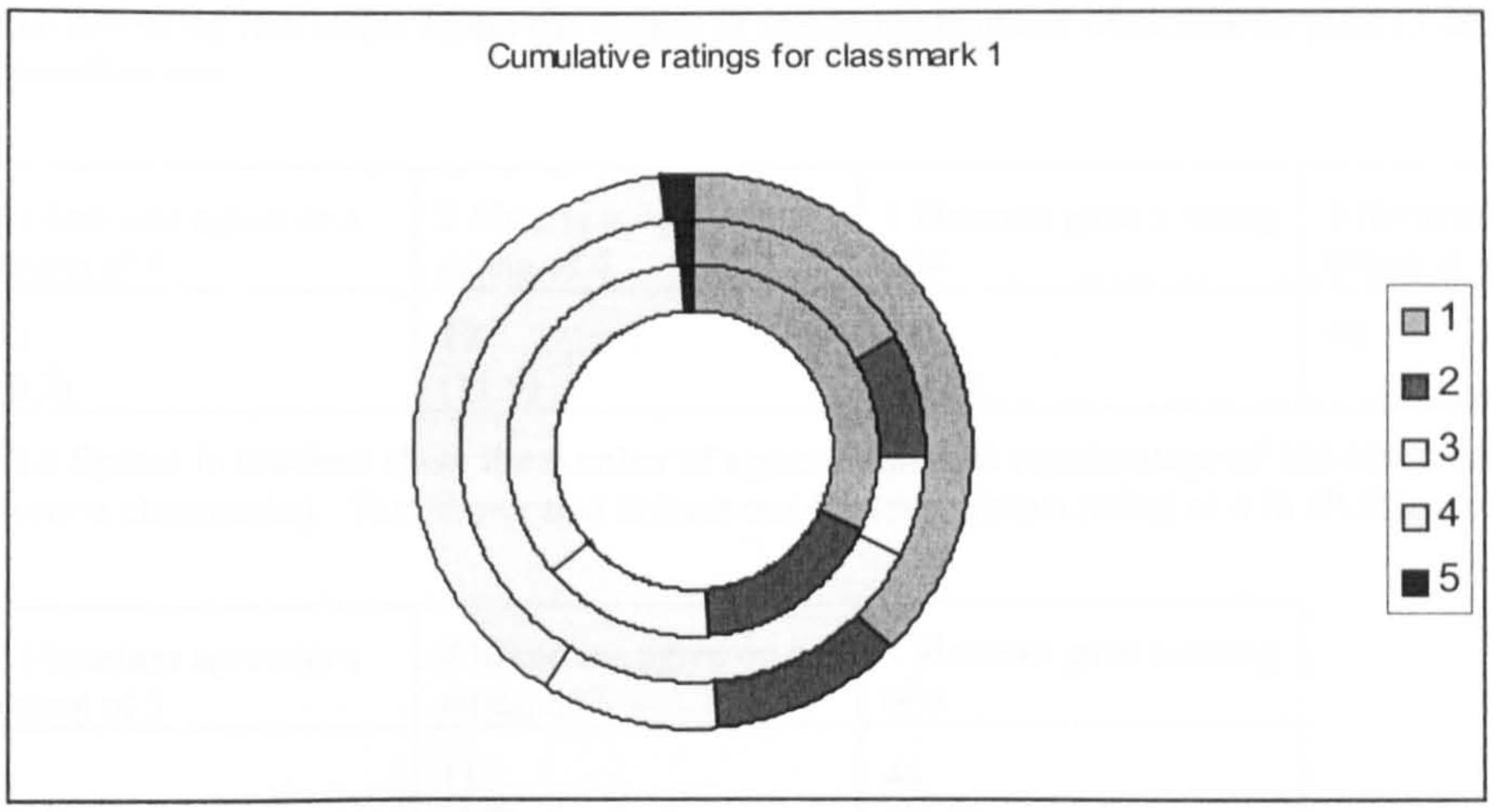


Figure 40. Doughnut chart showing the cumulative ratings for classmark 1

From these cumulative results it can be concluded that the classifier does show a level of effectiveness with 57.6% of the classifications acquiring a good (3 or 4) rating. This is considerably higher than the results first reported by the original Old ACE classifier (Burden and Wallis, 1996). Taking into account the degree of subjectivity in assessing the value of a classification this result is encouraging. Clearly there was some degree of variability in the ratings of the librarians. Librarian 2, for example, gave 73% of the first classifications a good rating of 3 or 4.

Some interesting observations can be made concerning different people's perceptions on the important aspects of information. For example, page number 86 linked from the results page is a page that lists degree courses in Artificial Intelligence. The classifier has given this page two classmarks:

classmark 1 - 150 Psychology
classmark 2 - 005.1 Programming (Computer Programming)

Librarians 2 and 3 are agreed on what the actual classmark here should be - 378 Higher Education. However, they disagree quite strongly on the rating of the automatic classifications: Librarian 2 has given both classmark 1 and classmark 2 a rating of 4, whereas librarian 3 has given both classmarks a rating of 1. Librarian 1 has taken a different approach altogether and has assigned the classmark 006.30711 (a convoluted subclass of 006.3 Artificial Intelligence) and has given the automatically assigned classmark 1 a rating of 2 and classmark 2 a rating of 3.

4.4.2.2 Cumulative Results for classmark 2

The following table shows the cumulative results for classmark 2.

Rating	Librarian 1	Librarian 2	Librarian 3	Total	%
1	86	54	104	244	49.2
2	21	19	18	58	11.7
3	34	34	13	81	16.3
4	24	58	26	108	21.8
5*	0	0	4	4	0.8
Total	*165*	*165*	*165*	*495*	*100*

* Not rated due to insufficient information.

This shows that overall just 38.1% of the second classifications were considered good acquiring ratings of 3 and 4. 49.2% of the classifications acquired a rating of 1, 11.7% a rating of 2, 16.3% a rating of 3 and 21.8% a rating of 4. 0.8% were not rated due to insufficient information. Figure 41 shows a bar chart representing this data.

The following two tables assess the degree of agreement between librarians on good (3 and 4) classifications.

3 librarians agree on a rating of 4	2 librarians agree on a rating of 4	1 librarian gave a rating of 4	0 librarians gave a rating of 4
11 (6.7)	19 (11.5)	37 (22.4)	98

The figures in brackets show the number of agreed ratings as a percentage of 165 (the total number of second classmarks). This shows that at least one librarian gave a rating of 4 in 40.6% of cases.

3 librarians agree on a rating of 3	2 librarians agree on a rating of 3	1 librarian gave a rating of 3
5 (3.0)	11 (6.7)	44 (26.7)

This shows that at least 2 librarians agree on a 'good' (3 or 4) rating in just 46 cases which is 27.8% of the time.

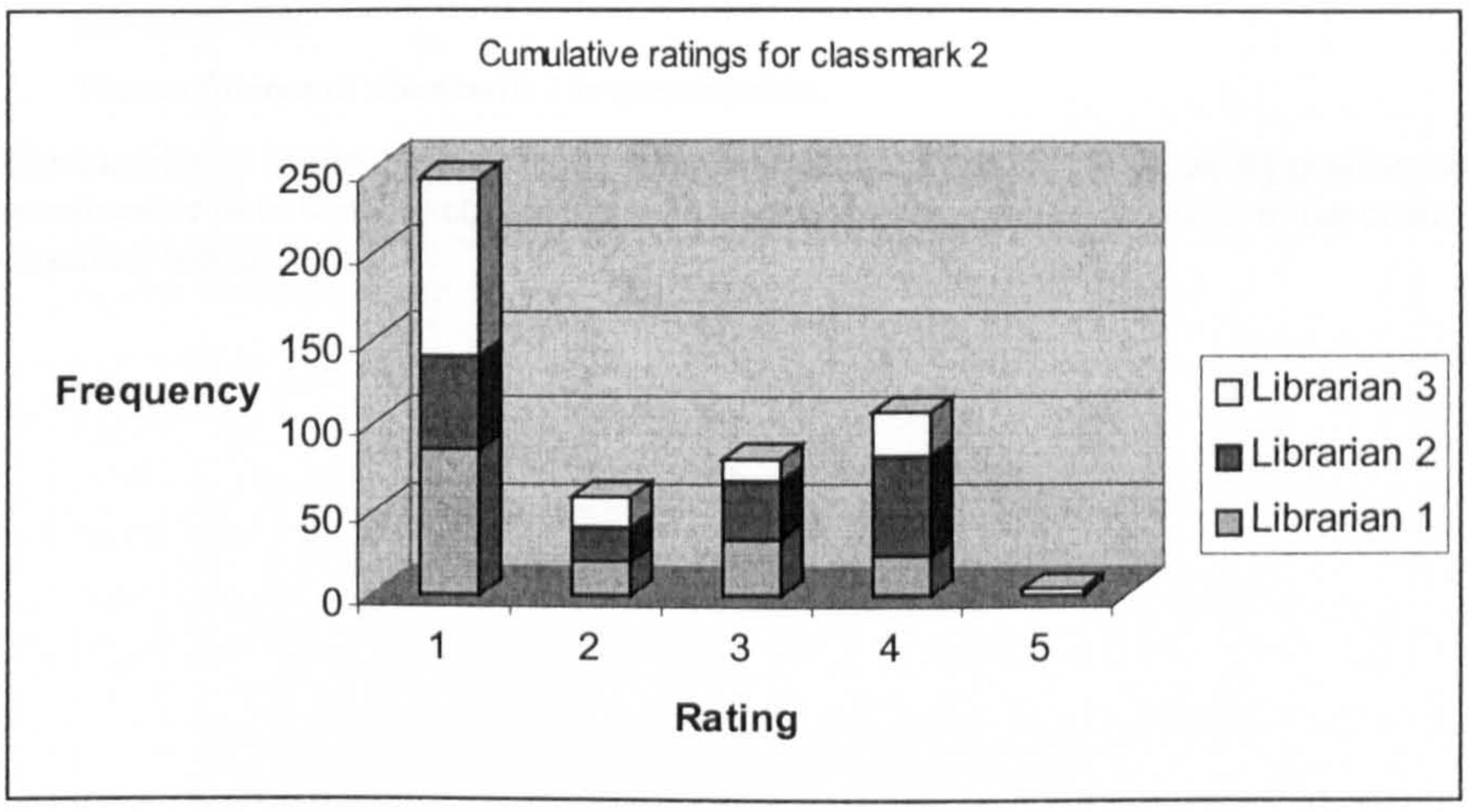


Figure 41. Bar chart showing cumulative ratings for classmark 2

Figure 42 shows a doughnut chart of the same data. The ratings for librarian 1 are represented by the inner ring, librarian 2 by the middle ring and librarian 3 by the outer ring.

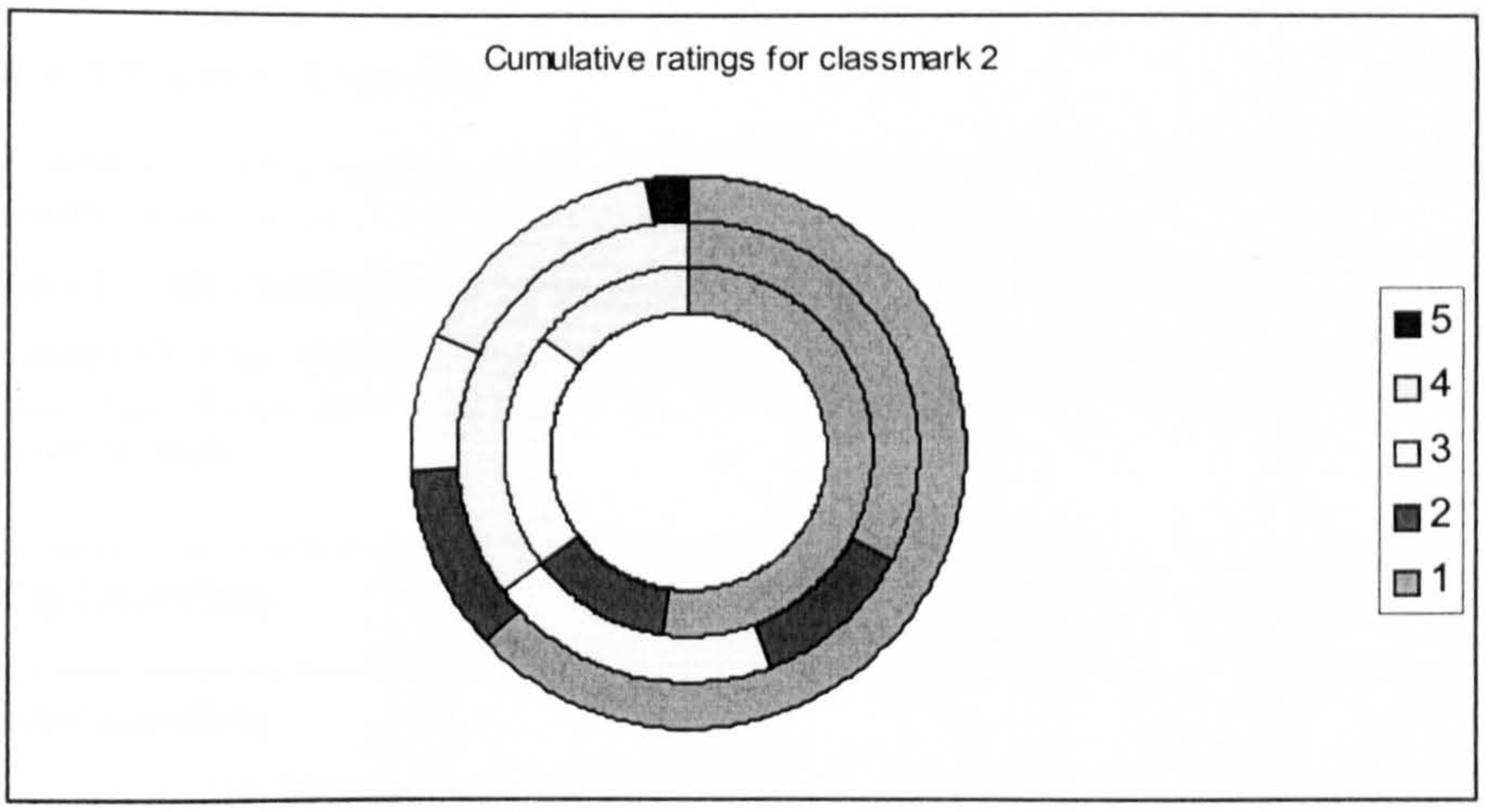


Figure 42. Doughnut chart showing cumulative ratings for classmark 2

The cumulative results from classmark 2 suggest two things:

1. The scoring and ranking of classmark objects appears to be effective - i.e. the best classification is presented first.
2. The usefulness of classmark 2 is questionable.

The cumulative results for the second classmarks showed just 38.1% acquiring positive ratings. It is questionable as to whether classmarks with such low ratings would be useful in the context of a classified web directory.

4.4.3 Further Analysis

Examination of the results spreadsheet in appendix J can reveal further interesting patterns in the results. In this section several questions are answered using this data:

4.4.3.1 Is the classifier more accurate in some areas of DDC than others?

Column O of the spreadsheet counts the total number of 4 ratings given to the first classification for each page. By sorting the data according to this column in decreasing order, the following analysis could be made:

Top Level Class	Total classifications	4 rating by 3 librarians	4 rating by 2 librarians	4 rating by 1 librarian	Total 4 ratings	No 4 ratings
000 Generalities	49	6 (12.2)	15 (30.6)	19 (38.8)	40 (81.6)	9 (18.4)
100 Phil/Psychology	2	0	0	2 (100)	2 (100)	0
200 Religion	1	0	0	1 (100)	1 (100)	0
300 Social Sciences	64	23 (35.9)	12 (18.8)	15 (23.4)	50 (78.1)	14 (21.9)
400 Languages	0	0	0	0	0	0
500 Natural Sciences	12	4 (33.3)	0	4 (33.3)	8 (66.7)	4 (33.3)
600 Technology	39	5 (12.8)	3 (7.7)	6 (15.4)	14 (35.9)	25 (64.1)
700 The Arts	31	15 (48.4)	5 (16.1)	4 (12.9)	24 (77.4)	7 (22.6)
800 Literature	2	0	0	1 (50)	1 (50)	1 (50)

It should be noted that there are no classifications falling into the top level 900 History and Geography class. The classifier does not support this class - no class representatives were developed on geography or history for this prototype.

The 'Total classifications' column, in the above table, indicates the total number of first classifications falling within each of the top-level classes. 49 of the randomly selected 200 classifications were assigned classmarks beginning with a 0 (from Generalities), 2 were assigned classmarks beginning with a 1 (from Philosophy and Psychology) and so on. The '4 rating by 3 librarians' column indicates how many of the classifications falling within each top-level class acquired a rating of 4 from all three librarians. The figure in brackets shows this frequency as a percentage of the total number of classifications in this class. The '4 rating by 2 librarians' column shows how many acquired a 4 rating from two of the three librarians and so on. The last two columns 'Total 4 ratings' and 'No 4 ratings' show how many got at least one 4 rating against the number that got no 4 ratings at all. So, for example out of the 49 classifications that fell within the generalities class (beginning with 0) 40 acquired at least one 4 rating and 9 got no 4 ratings at all. These two columns are compared graphically in the bar chart shown in figure 43.

The data from the above table shows that:

- 81.6% of the classifications falling into the Generalities top level class (000) acquired at least one 4 rating.

- 100% of the classifications falling into the Philosophy and Psychology top level class (100) acquired at least one 4 rating. This could, however, be misleading because there were only two classifications falling into this class in total and in both cases the 4 rating came from just one librarian.
- Similarly 100% of the classifications falling into the Religion top level class (200) acquired at least one 4 rating but there was in fact just one classification falling into this class and again the 4 rating came from just one librarian.
- 78.1% of the classifications falling into the Social Sciences class (300) acquired at least one 4 rating.
- No classifications fell into the Languages top level class (400).
- 66.7% of the classifications falling into the Natural Sciences class (500) acquired at least one 4 rating.
- 35.9% of the classifications falling into the Technology class (600) acquired at least one 4 rating.
- 77.4% of the classifications falling into the Arts class (700) acquired at least one 4 rating.
- 50% of the classifications falling into the Literature class (800) acquired at least one 4 rating. This could be misleading because just two classifications fell into this class and one of those acquired a 4 rating from just one librarian.

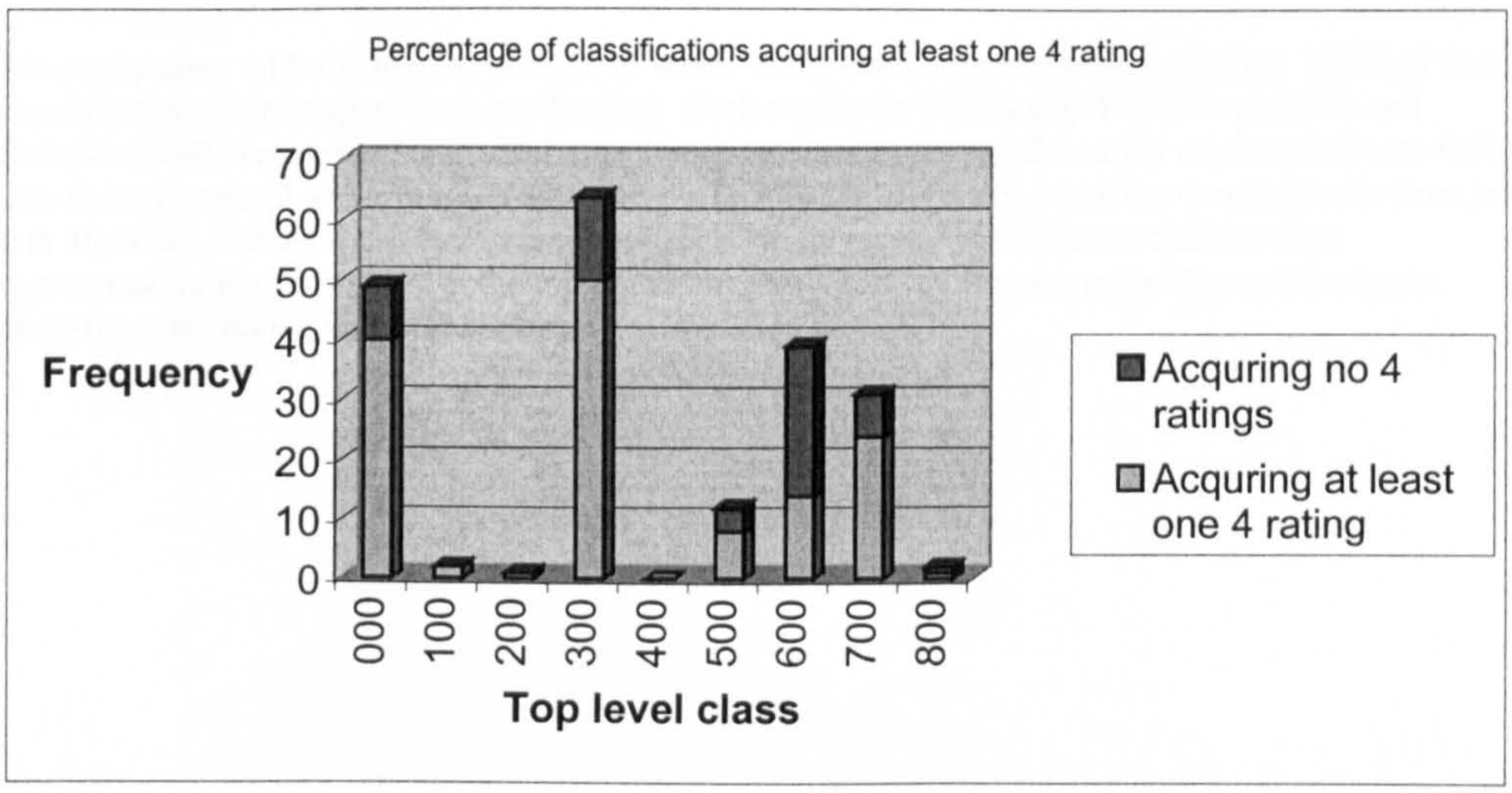


Figure 43. Showing the percentage of first classifications in each top level class acquiring at least one 4 rating

One possibility is that the length of the class representatives has an effect on the degree of accurate classifications. The table below shows class representative lengths for each of the top level classes against the percentage of classifications falling into each class that acquire at least one 4 rating:

Top level class name	Number of words in top level class representative	% of classifications acquiring at least one 4 rating
000 Generalities	173	81.6
100 Philosophy/Psychology	246	100
200 Religion	178	100
300 Social Sciences	116	50
400 Languages	16	0
500 Natural Sciences	93	66.7
600 Technology	168	35.9
700 The Arts	167	77.4
800 Literature	14	50

The two classes with the highest number of words in the class representative both show 100% of their classifications acquiring at least one 4 rating. Both results for Philosophy/Psychology (100) and Religion (200) were considered unreliable though because of the small number of classifications falling into these classes (2 and 1 respectively) and the fact that in all (three) cases the 4 rating came from just one librarian. Taking these factors into account it would appear inconclusive that the class representative length, at least at the top of the hierarchy, has any impact on the degree of accurate classifications falling within that class.

The same analysis was carried out using column P of the spreadsheet, which counts the number of 3 ratings given to the first classification for each page:

Top Level Class	Total classifications	3 rating by 3 librarians	3 rating by 2 librarians	3 rating by 1 librarian	Total 3 ratings
000 Generalities	49	1 (2)	4 (8.2)	12 (24.5)	17 (34.7)
100 Phil/Psychology	2	0	0	1 (50)	1 (50)
200 Religion	1	0	0	0	0
300 Social Sciences	64	1 (1.6)	1 (1.6)	14 (21.9)	16 (25)
400 Languages	0	0	0	0	0
500 Natural Sciences	12	0	1 (8.3)	2 (16.6)	3 (25)
600 Technology	39	0	2 (5.1)	5 (12.8)	7 (17.9)
700 The Arts	31	0	1 (3.2)	6 (19.4)	7 (22.6)
800 Literature	2	0	0	1 (50)	1 (50)

Note that in this instance it is not appropriate to consider the number that received no 3 ratings since they could have received up to three higher ratings of 4.

The following table combines the 'good' 3 and 4 ratings acquired from at least 2 librarians by the classifications falling in each class:

Top Level Class	Total classifications	Good ratings by at least 2 librarians	Less than 2 good ratings
000 Generalities	49	26 (53)	23 (46.9)
100 Phil/Psychology	2	0	2 (100)
200 Religion	1	0	1 (100)
300 Social Sciences	64	37 (57.8)	27 (42.2)
400 Languages	0	0	0
500 Natural Sciences	12	5 (41.7)	7 (58.3)
600 Technology	39	10 (25.6)	29 (74.4)
700 The Arts	31	21 (67.7)	10 (32.3)
800 Literature	2	0	2 (100)

This is a potentially more accurate picture showing that 67.7% of the classifications falling within the Arts (700) top level class acquire 'good' ratings from at least two librarians. The second highest is Social Sciences (300) where 57.8% of the classifications acquire good ratings from at least two librarians. One possible reason for these classes acquiring higher ratings could be that documents falling into these classifications are more verbose. Further research would need to be carried out to prove this. The low accuracy and low number of classifications recorded for Languages (400) and Literature (800) is likely to be a consequence of poorly defined class representatives (see previous table showing class representative lengths). The low figures for Religion (200) is less easily explained but could be due to low representation within the test data.

Figure 43 shows a graphical bar chart representation of this information.

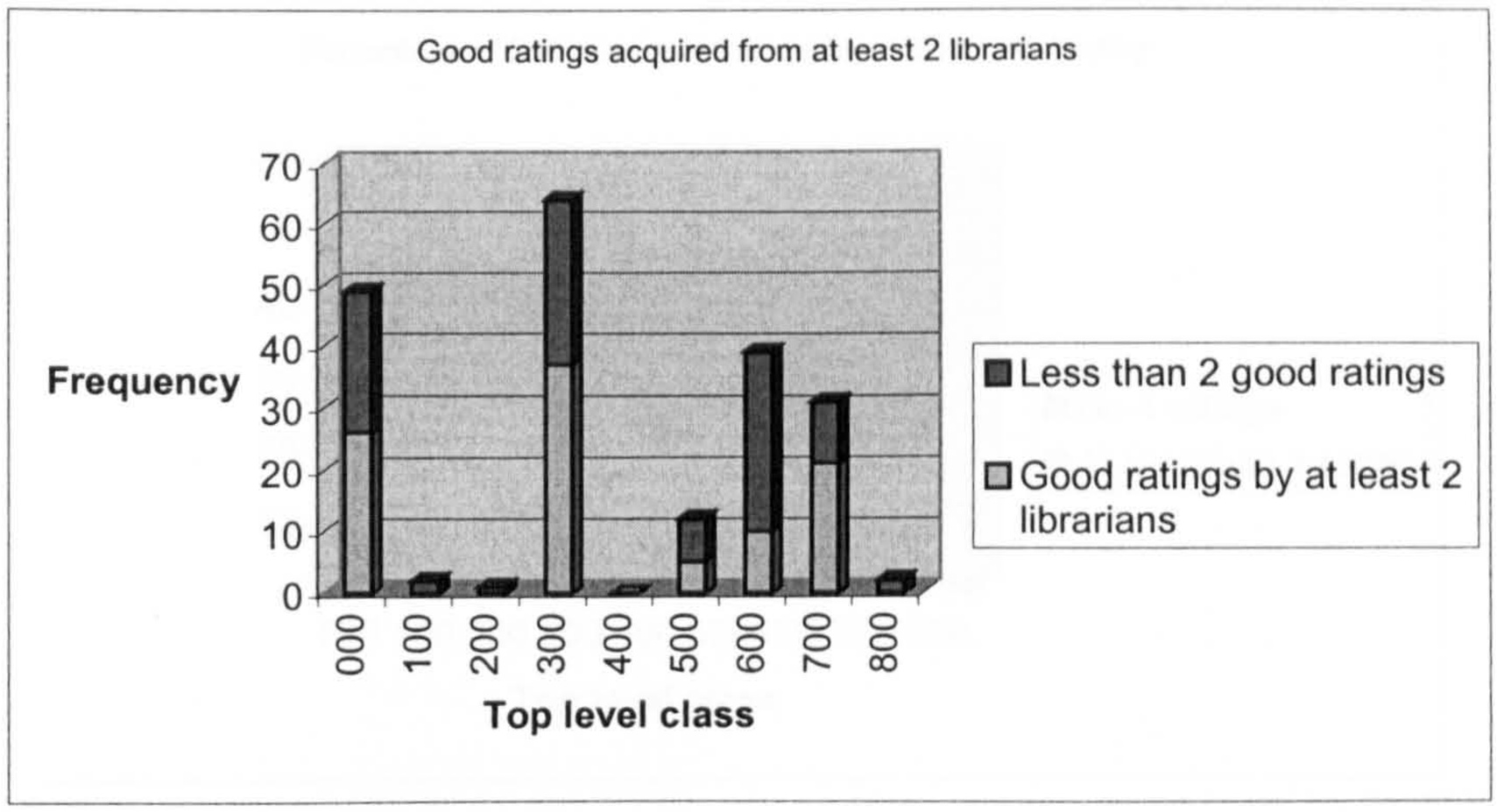


Figure 44. Showing percentage of good (3 and 4) ratings acquired from at least 2 librarians for classifications falling into each of the top level classes

The following table looks at the number of 4 ratings acquired by the second classmark for each page using column S of the spreadsheet:

Top Level Class	Total classifications	4 rating by 3 librarians	4 rating by 2 librarians	4 rating by 1 librarian	Total 4 ratings	No 4 ratings
000 Generalities	56	3 (5.4)	14 (25)	12 (21.4)	29 (51.8)	27 (48.2)
100 Phil/Psychology	5	0	0	0	0	5 (100)
200 Religion	2	0	0	1 (50)	1 (50)	1 (50)
300 Social Sciences	43	4 (9.3)	3 (7)	11 (25.6)	18 (41.9)	25 (58.1)
400 Languages	3	0	0	0	0	3 (100)
500 Natural Sciences	6	1 (16.7)	0	3 (50)	4 (66.7)	2 (33.3)
600 Technology	26	3 (11.5)	1 (3.8)	3 (11.5)	7 (26.9)	19 (73)
700 The Arts	24	1 (4.2)	1 (4.2)	7 (29.2)	9 (37.5)	15 (62.5)
800 Literature	0	0	0	0	0	0

Based on the information shown in the table, figure 45 shows the percentage of 2nd classmarks acquiring at least one 4 rating based on the above table.

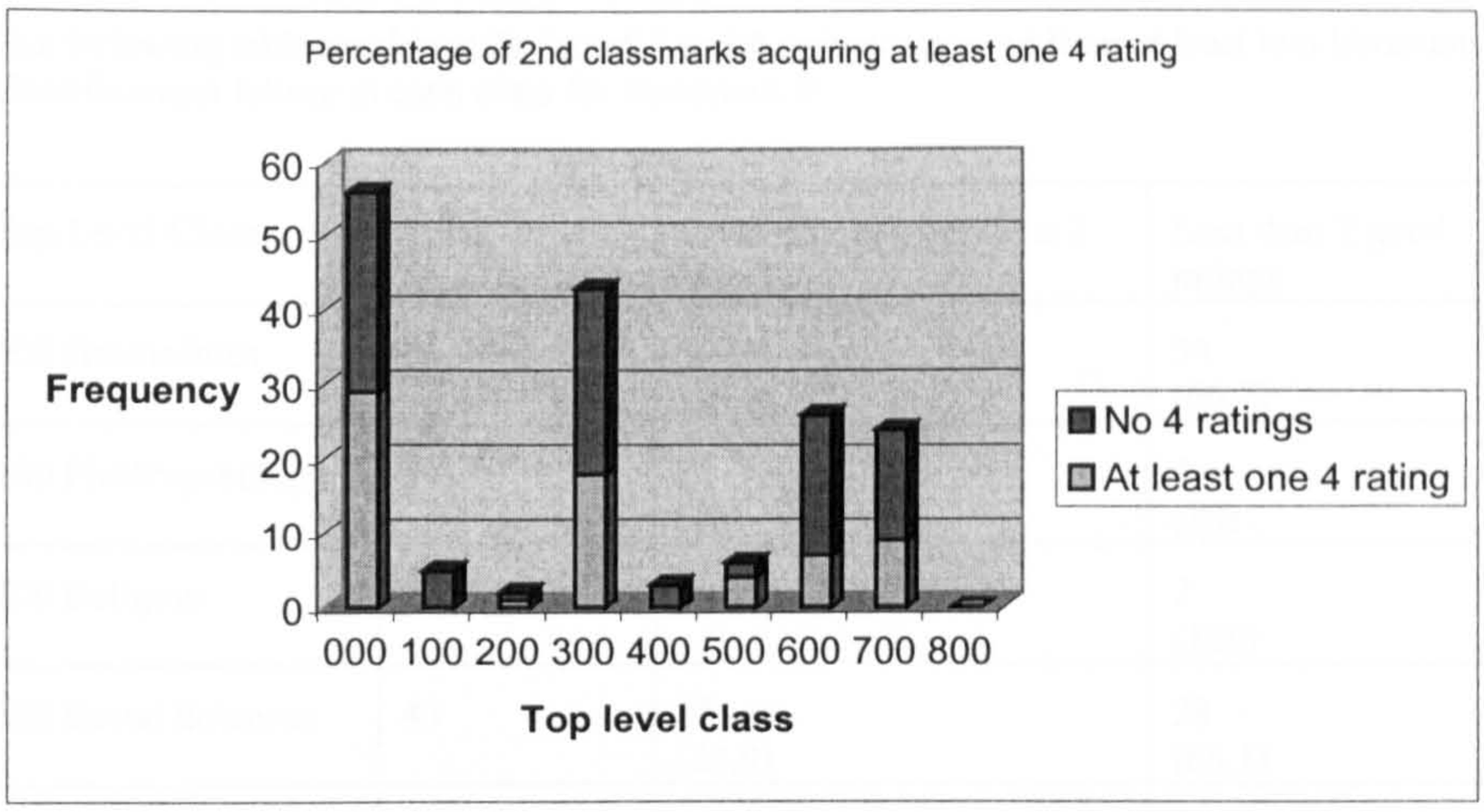


Figure 45. Percentage of 2nd classmarks acquiring at least at least one 4 rating

The same analysis was carried out using column T of the spreadsheet, which counts the number of 3 ratings given to the second classification for each page:

Top Level Class	Total classifications	3 rating by 3 librarians	3 rating by 2 librarians	3 rating by 1 librarian	Total 3 ratings
000 Generalities	56	1 (1.8)	4 (7.1)	22 (39.3)	27 (48.2)
100 Phil/Psychology	5	1 (20)	0	2 (40)	3 (60)
200 Religion	2	0	0	0	0
300 Social Sciences	43	2 (4.7)	6 (14.4)	6 (14.4)	14 (32.6)
400 Languages	3	0	0	0	0
500 Natural Sciences	6	0	0	1 (16.7)	1 (16.7)
600 Technology	26	0	0	7 (26.9)	7 (26.9)
700 The Arts	24	2 (8.3)	2 (8.3)	5 (20.8)	9 (37.5)
800 Literature	0	0	0	0	0

Again the number that received no 3 ratings were not counted in the above table because they may have received up to three higher ratings of 4.

The following table combines the 'good' 3 and 4 ratings acquired from at least two librarians by the classifications falling in each class for classmark 2:

Top Level Class	Total classifications	Good ratings by at least 2 librarians	Less than 2 good ratings
000 Generalities	56	22 (39.2)	34 (60.7)
100 Phil/Psychology	5	1 (20)	4 (80)
200 Religion	2	0	2 (100)
300 Social Sciences	43	15 (34.9)	28 (65.1)
400 Languages	3	0	3 (100)
500 Natural Sciences	6	1 (16.7)	5 (83.3)
600 Technology	26	4 (15.4)	22 (84.6)
700 The Arts	24	6 (25)	18 (75)
800 Literature	0	0	0

Figure 46 shows a graphical bar chart representation of this information.

The percentage of classifications from each top level class acquiring at least one 4 rating is clearly lower than for classmark 1. This is unsurprising since the cumulative results in the previous section showed that the ratings were lower generally for classmark 2 than for classmark 1. There are, however, some interesting consistencies in the classes that have acquired the highest and lowest percentage of 'good' ratings. The three classes acquiring the highest percentage of good ratings for classmark 1 were the Arts (700), Social Sciences (300) and Generalities (000) these three classes (in reverse order) also acquired the highest number of good ratings for classmark 2. Among the classes acquiring the lowest percentage of good ratings, were Religion (200), Languages (400) and Literature (800) all three of which acquired zero good ratings from more than one librarian for both classmark 1 and classmark 2.

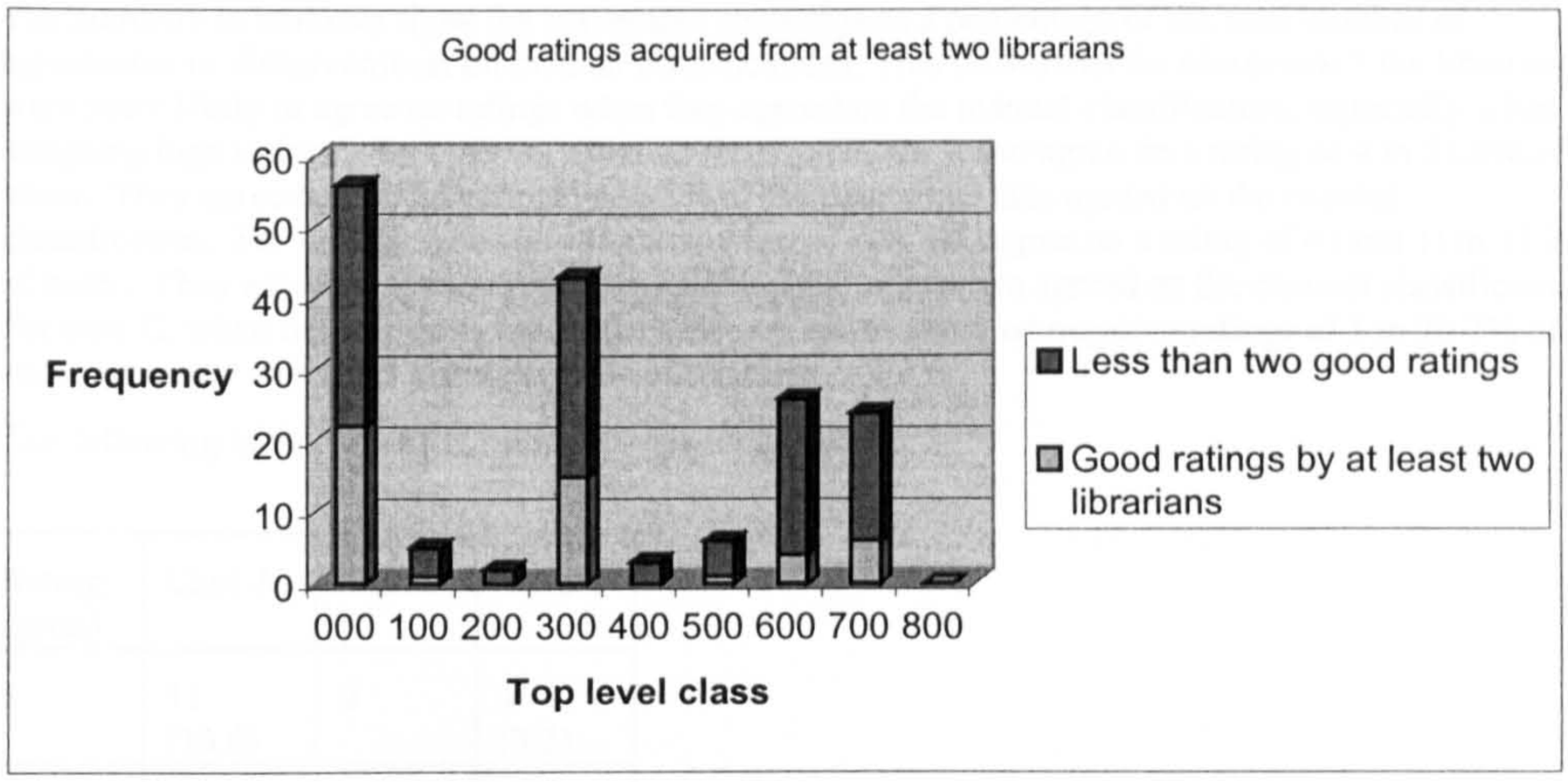


Figure 46. Bar chart showing percentage of good ratings from at least two librarians

From this analysis it can be concluded that the classifier has performed more accurately in some areas of DDC than others. The poor performance in some areas could be due to poorly defined class representatives and/or poor representation of the subject area among the corpus of test pages.

4.4.3.2 Do the librarians agree with each other on the rating when they agree on the manual classification?

This question can be answered by examining column W on the spreadsheet, which counts the number of common manually assigned classmarks for each page. This reveals that all three librarians agree just 81 times out of 200 on the manual classification of the pages as shown in the following table:

Number of librarians in agreement on manual classification	Number of classifications
3 librarians agree (Case A)	81
2 librarians agree (Case B)	89
0 librarians agree (Case C)	30

The following table shows how many times the librarians agree on each rating for classmark 1 at the same time as agreeing on the manual classification:

Rating agreed	Case A	Case B	Case C
4	42 (51.9)	10 (11.2)	1 (3.3)
3	2 (2.5)	0	0
2	0	0	0
1	8 (9.9)	10 (11.2)	8 (26.7)
Totals	*52 (64.2)*	*20 (22.5)*	*9 (30)*

The numbers in brackets show the agreement on ratings as a percentage of the total number of agreements or disagreements on manual classifications. This shows that for classmark 1 the librarians were more likely to agree on ratings when they agreed on the manual classification, especially when assigning high ratings. For case A, when all three agree, they also agree on a rating of 4 in 51.9% of cases. They agree on ratings generally 64.2% of the time when also agreed on the manual classification. For case B, when two librarians agree, they also agree on a rating of 4 (and 1) in 11.2% of cases. They agree on ratings generally 22.5% of the time when agreed on the manual classification. For case C, when none of them agree, they are inclined to agree on negative ratings of 1 in 26.7% of cases, agreeing on ratings generally 30% of the time.

The following table shows the same analysis for classmark 2:

Rating agreed	Case A	Case B	Case C
4	11 (13.6)	0	1 (3.3)
3	5 (6.2)	0	1 (3.3)
2	0	0	1 (3.3)
1	15 (18.5)	20 (22.4)	8 (26.7)
Totals	*31 (38.3)*	*20 (22.5)*	*11 (36.7)*

This analysis shows that for classmark 1, when the ratings were higher generally, in cases where the librarians agreed on the manual classification they also tended to agree on the rating, especially high ratings. Where they disagreed on the manual classifications, if they agreed on a rating at all it was a negative one. Analysis of the generally lower ratings, for classmark 2, is less conclusive. It does suggest, however, that in common with classmark 1, if all librarians disagree on the manual classification, where they do agree on a rating it is a negative one.

4.4.3.3 Do correct classifications tend to have higher scores?

This can be assessed by, firstly, sorting the results according column O that counts the number of 4 ratings for classmark 1, and then by sorting according to column C which provides the actual scores assigned by the classifier for each first classmark. This will give the range of scores for particularly good classifications. Then the opposite can be achieved by sorting the results according to column R which counts the number of 1 ratings for classmark 1. This will give the range of scores for particularly bad classifications. The result of this analysis is as follows:

Score Range	Frequency in good (4 rating) classifications	Frequency in bad (1 rating) classifications
0-1	0	0
1-2	6 (11.3)	4 (15.4)
2-3	5 (9.4)	7 (26.9)
3-4	4 (7.6)	2 (7.7)
4-5	12 (22.6)	5 (19.2)
5-6	3 (5.7)	0
6-7	6 (11.3)	5 (19.2)
7-8	3 (5.7)	1 (3.9)
8-9	4 (7.6)	1 (3.9)
9-10	3 (5.7)	0
10-11	1 (1.9)	0
11-12	0	0
12-13	2 (3.8)	0
13-14	1 (1.9)	1 (3.9)
14-15	2 (3.8)	
15-16	0	
16-17	1 (1.9)	
Totals	*53*	*26*

The same calculations can be acquired for classmark 2 by sorting according to column S which counts the number of 4 ratings for classmark 2 and then by column E which provides the actual scores assigned by the classifier for each second classmark. This will give the range of scores for particularly good classifications. Then the opposite can be achieved by sorting according to column V which counts the number of 1 ratings for classmark 2:

Score Range	Frequency in good (4 rating) classifications	Frequency in bad (1 rating) classifications
0-1	1 (4.4)	3 (7)
1-2	5 (21.7)	15 (33.9)
2-3	2 (8.7)	13 (30.2)
3-4	1 (4.4)	7 (16.3)
4-5	12 (52.2)	2 (4.7)
5-6	2 (8.7)	0
6-7	0	1 (2.3)
7-8	0	0
8-9	0	2 (4.7)
Totals	*23*	*43*

From the above, it would appear that there is no correlation between high ratings and higher scores. This is unsurprising since the score in each case will depend largely on the content of the particular document, the frequency of words and the appearance of certain words within particular HTML tags. Comparisons between scores given to classmarks assigned to the same document yields meaningful results - as indicated by the fact that the highest scoring classmarks (classmark 1s) have proved to be more effective than the second highest scoring (classmark 2s) for each document. Scores for classmarks are not necessarily relative between documents however. The nature and structure of each individual document will effect the scores given to the classmarks assigned to it. This is one of the reasons why a threshold approach was applied to the consideration of scores when filtering documents through the classification hierarchy (as described in section 3.2.3.4).

4.5 Summary

Firstly, to summarise the positive ratings from each librarian for classmark 1:

Librarian number:	1	2	3
% Good (3 and 4) ratings:	50.5	73	49.5

The cumulative effects of these ratings are that 57.6% of the first classifications (classmark 1) were considered good, acquiring ratings of 3 and 4. It was also found that at least 2 librarians agreed on a good rating in 49.5% of cases.

For classmark 2 the positive ratings from each librarians were as follows:

Librarian number:	1	2	3
% Good (3 and 4) ratings:	35.1	55	23.5

The cumulative effects of these ratings are that just 38.1% of the second classmarks (classmark 2) were considered good with at least 2 librarians agreeing on a good rating in just 27.8% of cases.

From these result is can be asserted that highest scoring classmarks (classmark 1) presented by the classifier show an encouraging level of effectiveness with 57.6% acquiring high ratings. The second

classmarks, acquiring just 38.1%, show a less encouraging level of effectiveness. This however, reflects the fact that the classifier successfully ranks the classmarks according to accuracy.

Further analysis showed that the classifier was more accurate in the Arts (700), Social Sciences (300) and Generalities (000). This was thought to be due to there being better defined (more wordy) class representatives in these areas, combined with a certain degree of poor representation among the test cases of other areas. Religion, for example, has reasonably well defined class representatives but there were few documents on Religion among the 200 randomly selected.

There was some evidence to suggest that librarians were more likely to agree on ratings when they agreed on manual classifications. For classmark 1, for example, in 64.2% of cases where they had agreed on a manual classification they also agreed on the rating. When they disagreed on the manual classification, if they agreed on the rating at all it was likely to be (in 26.7% of cases for classmark 1) on a negative rating of 1.

5. Metadata Generation

The World Wide Web Consortium (W3C 1999) has introduced the Resource Description Framework (RDF), in an attempt to produce a standard language for machine-understandable descriptions of resources on the Web. RDF is intended to support resource descriptions for resource discovery and also for rights management, privacy preferences, content ratings like PICS (Resnick 1998), evaluation and classification. RDF is seen as the framework for providing a *Web of trust* where the content of each individually accessible object is well described in a format that is extensible yet universally understood. Metadata describing a resource could be accredited by a third party organisation using RDF encoded digital signatures providing a reliable, trust-worthy authentication mechanism. RDF may enable search engines and other tools for resource discovery to exchange and share metadata. This chapter describes how the automatic classifier, discussed in chapters 3 and 4 can be used to automatically generate metadata in RDF format for use in describing HTML documents for the purposes of resource discovery.

Section 5.1 discusses the metadata elements that are both accessible to the classifier and useful in the context of WWLib. Section 5.2 introduces the Resource Description Framework (Swick, 1998). Section 5.3 describes the process of automatic metadata generation. Section 5.4 describes the on-line metadata generator and section 5.5 provides a summary of this chapter.

5.1 Metadata Elements

As discussed in chapters 1 and 2, metadata, and the automatic generation of it, are important issues for automated search engines such as WWLib-TNG. The classification process results in the production of a series of classmarks appropriate to describe a particular document. However, the process can easily be used to extract various other metadata elements that are useful to the search engine. During the parsing of the document, terms found in the title element are singled out as being important, these can easily be extracted as can those terms which match those found in the class representatives of appropriate DDC leaf nodes i.e. significant keywords. Keywords and descriptions found in existing META tags can be extracted.

Other useful metadata that is easily accessible is shown in Figure 47. This element set is based on those metadata elements that are accessible to the classifier and useful to a search engine. It is thought that these elements (Wolverhampton Core) are sufficient to uniquely identify the document, state where it can be found, provide a good indication of the subject matter and of how current both the actual information and its metadata are.

The most well known and widely used metadata element set for resource discovery is Dublin Core (Weibel and Miller 1999). Compliance with a recognised standard is advisable because it encourages interoperability and consistency between applications. Dublin Core has evolved from the Digital Library community and consequently not all of its elements are as well suited to the automated search engine domain as those defined in figure 47. There is, however a significant overlap and none of the Dublin Core elements are compulsory. RDF enables developers to tailor an element set to suit their application while still reusing appropriate standard elements defined elsewhere (see section 5.2).

Figure 48 compares the fifteen elements of Dublin Core with the elements defined in figure 47.

	Element	Description	Purpose
1	Unique accession number	Number assigned by the system.	Uniquely identifies the resource within WWLib-TNG.
2	Title	Taken from the HTML <TITLE> element.	Usually helps in discerning the subject matter.
3	URL*	The URL given to the system, used to extract the document for classification.	Indicates the location of the document.
4	Abstract	Either the first 25 words found in the body of the page, or, if present, taken from the Description META tag. (A much more sophisticated abstracting technique could be used here in future implementations).	Provides further clues about the subject matter.
5	Keywords	Terms found within the document that match terms found within the class representatives of DDC classes found to be appropriate.	Indicate key issues/topics.
6	Classmarks	DDC classmarks found to be appropriate as a consequence of the classification process.	Indicate subject area(s).
7	Word count	The number of words found on the page, including the title.	Indicates extent, detail, download time.
8	Classification date	The system date when the classification took place (GMT or BST)	Indicates currency of the metadata.
9	Last modified date when classified	Taken from the HTTP Last-modified header. (Gives Not known if equal to the "epoch" - 1st January 1970)	Indicates currency of the information.

* The classifier only handles individual HTML documents so the URL, not URI, is appropriate. The URL is not used as an identifier within the search engine because it is possible for the same page to have more than one URL; this is one of the causes of repetitions in automated search engine results.

Figure 47. An appropriate metadata element set - The " Wolverhampton Core "

	Dublin Core Elements	Equivalent Wolverhampton Core Elements
1	Title	Title
2	Creator	-
3	Subject	Keywords + Classmarks
4	Description	Abstract
5	Publisher	-
6	Contributor	-
7	Date	Last modified when classified
8	Type	-
9	Format	-
10	Identifier	Accession number + URL
11	Source	-
12	Language	-
13	Relation	-
14	Coverage	-
15	Rights	-
16	-	Date Classified
17	-	Word count

Figure 48. Comparison between Dublin Core and the Wolverhampton Core element sets.

It can be observed that most of the Wolverhampton Core elements have a Dublin Core equivalent. The implications of this comparison are discussed again in the later section (5.2.2) on RDF schema definition. It is thought that the specified Wolverhampton Core elements represent an appropriate subset of Dublin Core (with one or two additions) that is suited to the requirements of an automated search engine.

Once the necessary metadata elements have been identified they can then be represented in RDF.

5.2 The Resource Description Framework (RDF)

RDF (Swick 1998) is intended to transform information on the Web from being machine-readable to machine-understandable. RDF provides the foundation on which machine-understandable resource descriptions can be exchanged and processed in an interoperable manner. RDF is intended to highlight the important aspects of a resource prior to automatic processing.

Three things are required in order to generate RDF statements about a resource: a data model, a schema and the actual representation in XML (eXtensible Markup Language (Connolly and Bosak 1998)) syntax. Several RDF schemas might actually be involved; schemas are required for the interpretation of RDF statements. The following three subsections explain how the metadata elements shown in figure 47 can be represented by an RDF data model, defined using an RDF schema and, most importantly, automatically generated in RDF/XML syntax.

5.2.1 RDF Data Model

The RDF data model provides the notation for representing properties and values. Properties represent the attributes of a resource and as such the data model represents the traditional name-value pairs associated with a resource as shown in figure 49. The RDF data model is expressed using directed labelled graphs (or "nodes and arcs" diagrams) which identify the properties and property values

associated with a resource (This notation is taken from the RDF Model and Syntax Specification (Lassila and Swick 1998)).

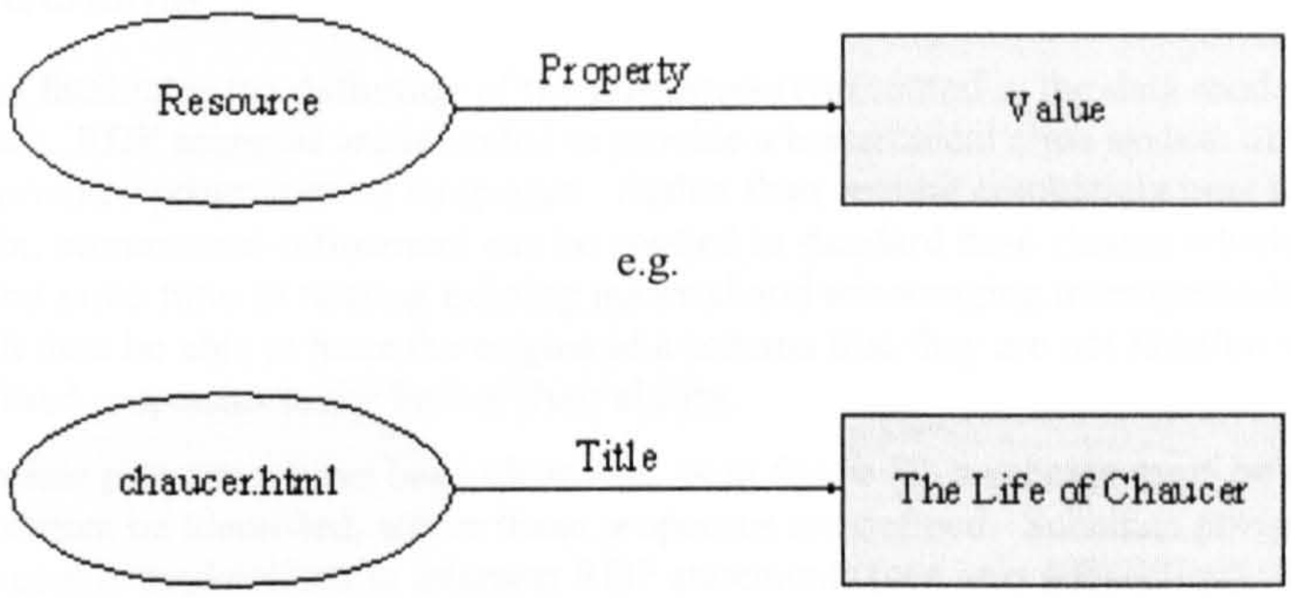


Figure 49. Data Model notation showing an RDF statement; a resource, a named property and the value of that property.

In RDF a resource may be a simple Web page, part of a simple Web page, a collection of pages or a whole Web site. The automatic metadata generator described here generates descriptions of individual HTML pages.

Figure 50 shows a data model for the Wolverhampton Core elements:

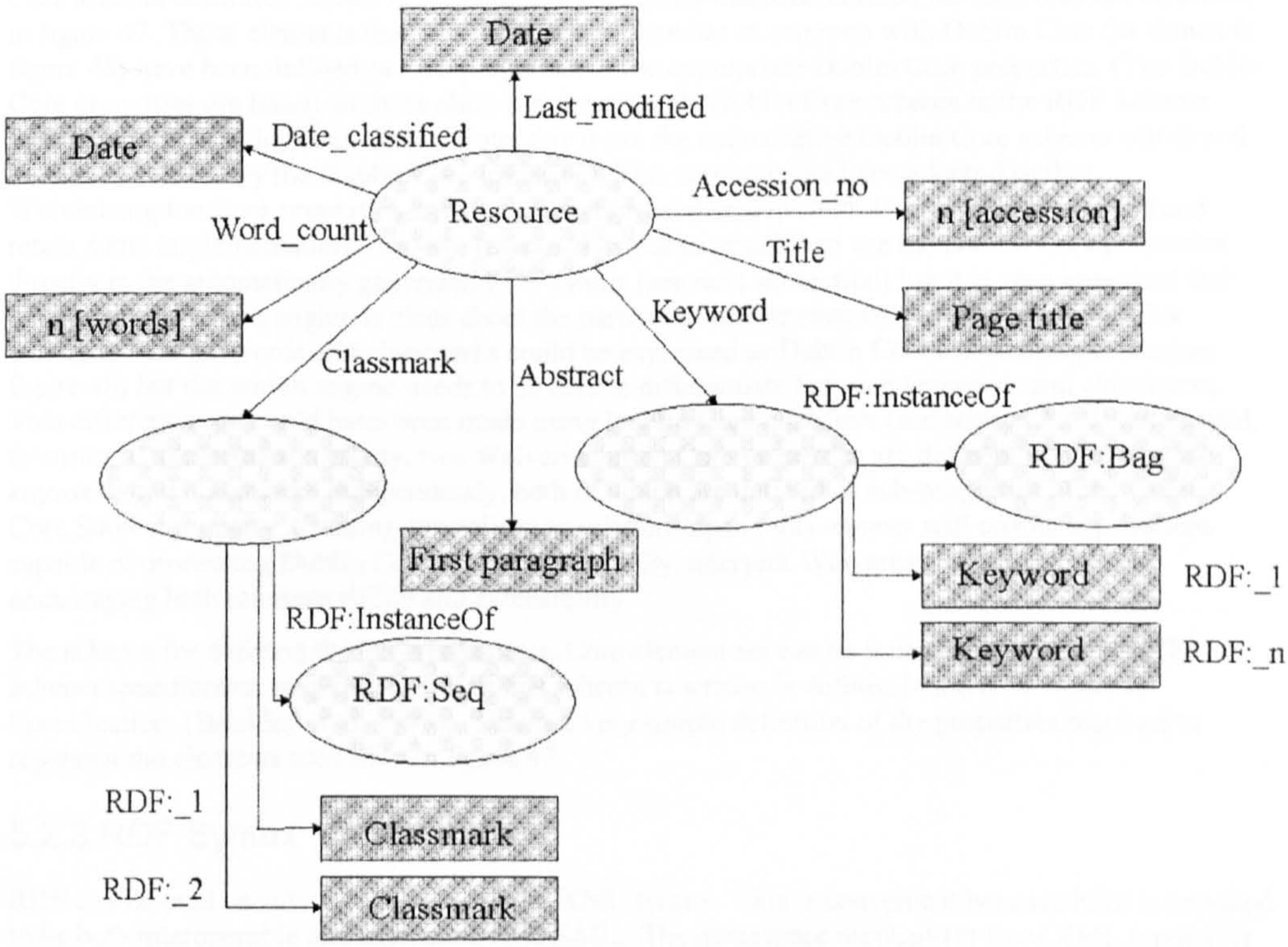


Figure 50. RDF Data Model for the Wolverhampton Core Elements

The model shows two RDF containers - one a bag of keywords and the other a sequence of classmarks. The classification process will usually result in the identification of several keywords within the document but the order in which they are presented is insignificant so a bag is appropriate. A better method of representing the keywords would be to use a Set where no duplicates would be permitted, however, RDF does not define a Set because there is no defined enforcement mechanism in the event of violation. The classmarks are ordered by the classifier according to which scored the highest measure of similarity and so these are represented as an ordered sequence. The classmarks would be

better represented by an ordered collection class where no duplicates would be allowed. Further work layered on the RDF core may define such enforcement mechanisms.

5.2.2 RDF Schema

The RDF schema facilitates the definition of the properties represented in the data model and expressed in the RDF syntax. RDF schemas are intended to provide a hierarchical class system not unlike that found in object oriented programming languages. Rather than writing completely new classes for a particular domain, incremental refinement can be applied to standard base classes which enables extensibility at the same time as reusing existing material and encouraging interoperability. Applications will then be able to trace the origins of a schema that they are not familiar with in order to interpret the defined properties to the best of their ability.

Once the appropriate properties have been identified, as in figure 50, a schema must be created, or existing schemas must be identified, where these properties are defined. Schemas provide the RDF type system and enable applications to interpret RDF statements (see next subsection). The properties could be expressed using appropriate existing vocabularies, in which case it is not necessary to define a new schema - existing schemas can be referenced from within the RDF/XML syntax. It is possible to reference as many different schemas as required, mixing and merging different vocabularies. Schemas are referenced using the namespace mechanism from XML within the RDF syntax (see next subsection).

The definition of new schemas enables developers to specify properties best suited to their particular application. Schemas can define properties that are sub-properties (or refinements as described above) of those defined elsewhere in existing schemas. This feature has been utilised in the Wolverhampton Core schema definition shown in appendix K. A property has been defined for each element identified in figure 47. Those elements that Wolverhampton Core has in common with Dublin Core (as shown in figure 48) have been defined as sub-properties of the appropriate Dublin Core properties. (The Dublin Core properties are based on those shown in the example Dublin Core schema in the RDF Schema Specification (Brickley et al. 1998). Note, this is not the authoritative Dublin Core schema which will be made available by the Dublin Core Initiative). This approach has been adopted so that Wolverhampton Core properties have specialisation relationships with Dublin Core properties and retain some implementation freedom. It would have been possible to use the Dublin Core properties directly in the automatically generated RDF syntax (see next subsection) but it is very important that the automated search engine is clear about the particular implementation of these properties. For example, both keywords and classmarks could be expressed as Dublin Core Subject properties (see figure 48) but the search engine needs to be able to differentiate between keywords and classmarks. This differentiation could have been made using Dublin Core qualifiers (see section 2.3.2) but instead, to ensure implementation clarity, two Wolverhampton Core properties are defined representing the keywords and classmarks independently, both of which are defined as sub-properties of the Dublin Core Subject property. Creating specialisation relationships in this manner will enable applications capable of processing Dublin Core to, at least partially, interpret Wolverhampton Core thereby encouraging both interoperability and extensibility.

The schema for defining the Wolverhampton Core element set can be found in Appendix K. (The schema specification language in which this schema is written is defined in the RDF Schema Specification (Brickley et al. 1998)). This is a very simple definition of the properties required to represent the elements identified in figure 47.

5.2.3 RDF Syntax

RDF can be (and usually is) expressed using XML syntax. This is convenient because RDF is intended to be both interoperable and extensible as is XML. The namespace mechanism from XML (xmlns) is used to specify the location of the schema that defines the properties expressed in the RDF description.

The following shows the RDF representation of the data model shown in figure 50 for the School of Computing and IT home page at the University of Wolverhmapton. Appendix L shows automatically generated RDF for a series of test URLs. (The RDF/XML syntax used here is described in The RDF Model and Syntax Specification (Lassila and Swick 1998).)

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
     <rdf:Description about="http://www.scit.wlv.ac.uk/">
          <wc:Accession_no>583295</wc:Accession_no>
          <wc:Title>SCIT WWW Server</wc:Title>
          <wc:Abstract>
                  School of Computing and Information Technology WWW
                  Server General Information University of
                  Wolverhampton School of Computing and Information
                  Technology home page Wolverhampton and surrounding
                  areas
          </wc:Abstract>
          <wc:Keyword>
               <rdf:Bag>
                    <rdf:li>computer</rdf:li>
                    <rdf:li>computing</rdf:li>
                    <rdf:li>database</rdf:li>
                    <rdf:li>databases</rdf:li>
                    <rdf:li>server</rdf:li>
                    <rdf:li>search</rdf:li>
                    <rdf:li>searching</rdf:li>
                    <rdf:li>directory</rdf:li>
                    <rdf:li>university</rdf:li>
                    <rdf:li>school</rdf:li>
               </rdf:Bag>
          </wc:Keyword>
          <wc:Classmark>
               <rdf:Bag>
                    <rdf:li>005.7 Data in computer systems</rdf:li>
                    <rdf:li>370   Education</rdf:li>
               </rdf:Bag>
          </wc:Classmark>
          <wc:Word_count>216</wc:Word_count>
          <wc:Last_modified>20/11/1997</wc:Last_modified>
          <wc:Classification_date>11/09/1998</wc:Classification_date>
     </rdf:Description>
</rdf:RDF>
```

Note that there are two XML namespace definitions (xmlns) at the top of this piece of RDF. The first one identifies the location of the RDF syntax specification and the second one identifies the location of the Wolverhampton Core (wc) schema where the property types specified within this RDF description are defined. This wc schema is shown in Appendix K.

W3C and the Dublin Core Initiative recommend the use of ISO 8601 Date format. This has not been implemented in this instance because the automatic metadata generator is to be deployed as part of a UK search engine where dates will be required in UK format.

5.3 Automatic RDF Metadata Generation

Configuring the classifier to extract and generate the metadata elements shown in figure 50 in RDF syntax, was a relatively simple task.

5.3.1 Title, Abstract, Word count and Accession number

As the classifier parses the document to generate a document object (shown in figure 14), the HTML tags are stripped out by the noHTML method of the Document class. During this process of stripping out the tags, markers are placed where the title (<TITLE></TITLE>) and heading (<Hn></Hn>) tags were found. Each line read from the file, or data input stream, is processed in this manner before being

tokenised and then each word found is added to the vector of keywords representing the document. When a title or heading marker is found, the weight stored with the proceeding words, until the end marker is found, is incremented accordingly. A Boolean variable called "Title" or "Heading" is set to true each time a marker is found and then set back to false when the end marker is encountered and the weight that is to be associated with each word is restored to the ordinary level. For the automatic metadata generating version of the classifier, each word found between the start and end markers for the title is also appended to a string variable which holds the title string after processing.

Similarly the automatic metadata generating version of the classifier is configured to identify meta tags and in particular the description and keywords tags. The words found in the keywords tag are just added to the keywords vector like any other word but with an incremented weight as a word in the title or heading would be. The words found in the description tag are also stored in this way but are simultaneously appended to a string variable that stores the abstract. If no meta tags are found (or at least, no description tag is found), the first 25 words from the body element are marked and appended to the abstract variable instead. Clearly a more sophisticated abstracting technique could be applied here but this was considered beyond the scope of this project.

After parsing, the automatic metadata generating version of the classifier has a document object (rdfdocument) which has two additional publicly accessible variables - titletext and abstracttext.

The word count is calculated by the document object as words are added to the keyword vector and it is always publicly accessible as it is used by the classify object when calculating the Dice Coefficient.

The accession number is usually stored and maintained by the document object, although it is not normally set (i.e. it is zero) when the classifier is running as a stand alone application. The accession number (set or otherwise) is however, publicly accessible from the document and rdfdocument objects.

rdfdocument.java can be found in appendix M.

5.3.2 Keywords and Classmarks

The automatic metadata generating version of the classify object (rdfclassify) generates classmarks as usual but also generates appropriate keywords. When the document is classified, the "proceed" method is called recursively until the document has been compared with all the appropriate branches of the DDC hierarchy. If the process has been successful, it results in the generation of a vector of at least one DDC classmark object, each of which holds within it a score representing the level of similarity between the class and the document. The classify object then works out which two of the selected classmarks (if there are more than one, which there usually are) are the highest scoring and these are presented as the two most appropriate classmarks. The rdfclassify object also, keeps track of significant keywords by appending successfully matched words to a string variable for a particular DDC class. If that class is then found to have a significant measure of similarity with the document, the string is appended to a string of "significant" words which is passed on as the recursive process proceeds through that branch of the hierarchy and if it leads to a significant match with a leaf node, those words are then stored with the classmark. If that class happens to be one of two highest scoring classes, the significant words for that class are identified as keywords for the document.

So the rdfclassify object has an additional publicly accessible string variable - sigkeywords - as well as the usual classmarks.

rdfclassify.java can be found in appendix M.

5.3.3 URL, Last modified date and Date classified

The automatic metadata generating version of the ACE object (rdface), has access to the URL or the path to the local file, as does the usual version of ACE, but also acquires the last modified date and current date when the classification takes place. The last modified date is acquired from the getLastModified method of the HttpURLConnection class in Java. If the date is not available, or it is a local file that is being classified, the string "Not Known" is shown. The current date is acquired from the Date class in Java.

5.3.4 RDF syntax output

The rdface object has an additional method called outputrdf where the appropriate metadata elements are acquired from the rdfdocument object, the rdfclassify object and the by rdface object itself (the source code classes for each of these objects can be found in appendix M) and these form the Wolverhampton Core properties expressed in RDF/XML as can be seen in the examples in appendix L and in the example in section 5.2.3 above.

5.4 On-Line Metadata Generator

The automatic metadata generator was available on-line during July 1999. The on-line version was implemented using a Java Servlet the source code of which can be found in appendix N. An HTML form (shown in figure 51) was used to acquire a URL from the user. The user could choose to see "Just the DDC Classmarks", "RDF using Dublin Core " or "RDF using Wolverhampton Core" describing the given resource. Hitting the "GENERATE" button resulted in the automatic generation of metadata in the requested format. Wolverhampton Core examples are shown in appendix L.

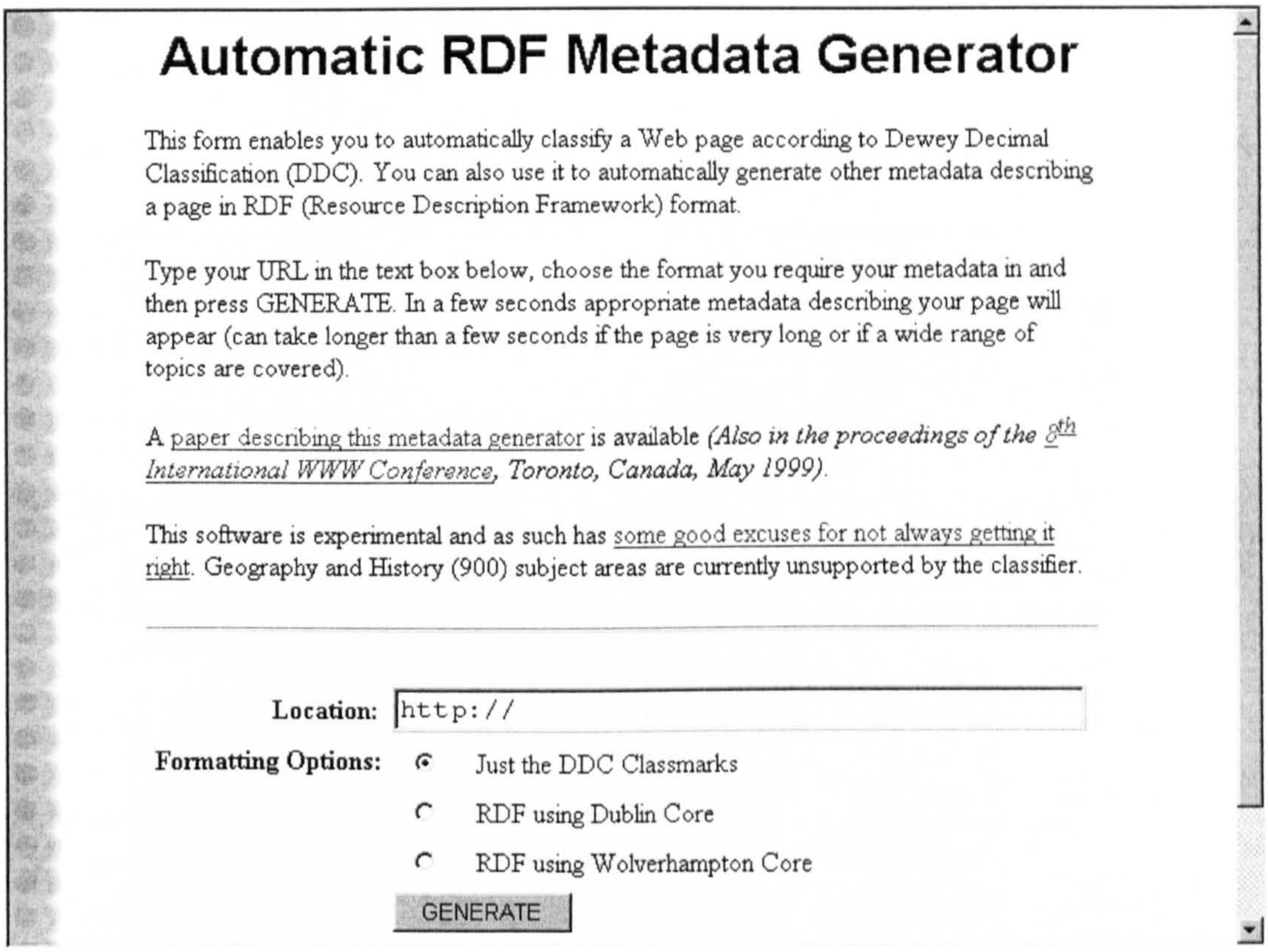


Figure 51. The on-line automatic metadata generator

This system was found to be much more reliable than the original client-server application developed for testing the classifier (shown in figure 51).

5.5 Summary

Although it is envisaged that the editing tools of the future will encourage the inclusion of RDF meta information, the current situation, where some authors choose not to include any metadata, is likely to continue to some extent. It is very difficult to automate resource description but it would be impossible to describe everything on the Web manually. Automatic metadata generation would appear to be an essential pre-requisite for widespread deployment of RDF based applications. The *Web of trust* must attempt to be comprehensive because a Web that is partially trust worthy offers little advantage over one that cannot be trusted at all, especially where content rating is concerned.

The automatic metadata generator described in this chapter enables an RDF description to be associated with any HTML page, regardless of when it was created and by which editing tool. RDF has enabled the specification of a metadata element set that is tailored to suit an automated search engine but strongly related to a standard, digital library element set, Dublin Core. The ability to create specialisation relationships with appropriate Dublin Core properties increases the potential for interoperability - any application capable of processing Dublin Core will be capable of processing most of the defined Wolverhampton Core properties because they are defined as sub-properties of Dublin Core properties. Such interoperability will encourage information sharing which will improve comprehensive Web coverage; if search engines can process the same standard syntax, they will be able to exchange metadata and integrate their results. Some subject-specific classified directories are known to be attempting to share information through the use of RDF already; information sharing between automated search engines has even greater potential.

6. Conclusions

The main aim of this project was to find out if it is possible to improve automated tools for resource discovery on the Web by enabling automatic classification and metadata generation. Reaching this aim involved:

- Investigation into the different approaches to information resource discovery on the Web.
- Investigation into the influence of traditional information retrieval (IR) and library science on web search engines.
- Exploration of the potential of automatic classification for automated search engines.
- Design and implementation of an automatic classifier.
- Evaluation of the classifier.
- Exploration into the possibility of automatically generating metadata for Web resources using the automatic classifier.

Various approaches to information resource discovery were found (see chapter 2); they generally fall into two categories:

1. Manually maintained tools - accurate, high quality information, well focused, classified, directory-style, often out of date and incomplete.
2. Automated tools - comprehensive coverage, more up to date, unclassified, poor quality information, no notion of context.

In conclusion it was decided that the use of automatic classification could perhaps 'bridge the gap' between these two approaches by enabling automated search engines to organise resources by subject. This would enable them to acquire some notion of context and would hopefully result in more focussed results at the same time as providing current, comprehensive Web coverage.

6.1 Contribution of this Project

A hierarchical classifier was designed and developed that classifies Web pages according to DDC. Some aspects of the classifier (described in detail in chapter 3) drew inspiration from traditional IR. However, some IR techniques were considered inappropriate for application in the Web domain due to the erratic, unwieldy nature of the data (Wong and Fu, 2000).

The classifier was extended to perform automatic RDF (Swick, 1998) metadata generation using an interoperable schema that was specially designed with the metadata requirements of WWLib in mind. This metadata generation is described in detail in chapter 5.

Four main contributions have been made by this project:

1. A novel methodology for automatically classifying Web pages according to DDC.
2. Design and implementation of a classifier which, as a stand alone application, could be used not only to enhance future implementations of WWLib but also other tools for information resource discovery on the web.
3. A schema defining metadata elements of particular importance to automated search engines.
4. Design and implementation of an automatic RDF metadata generator built around the automatic classifier described above.

The evaluation exercise (chapter 4) indicates that automatic classification according to DDC is possible and is potentially effective in improving the performance of a search engine. Classification can be used to improve selectivity by presenting the user with whole collections of documents associated with query terms in addition to individual documents containing the terms.

The automatic metadata generating aspect of the classifier was very successful. Resources that were accurately classified were also well described. Even those resources that were not accurately classified or those that were not classified at all were described in a useful fashion. The URL, title, abstract

(albeit a simple one), last modified date, date classified and word count were almost always available and accurate even if the classification classmarks and resulting keywords were not.

The integrated use of an automatic classifier within such an automatic metadata generator has considerable potential for automated search engines. Not only do the automatically generated descriptions dictate the clustering of documents sharing the same subject matter, but also the keywords (and potentially the abstract or summary) are generated in context. Obviously automated search engines already automatically generate metadata but not context sensitive, classified metadata.

The interoperable nature of the metadata generated is another important feature. The use of semantic and syntactic metadata standards (Dublin Core and RDF respectively) has enabled the generation of descriptions that could be interpreted by other search engines not just WWLib. According to Lawrence and Giles (1999) there is very little overlap in the Web coverage of the major search engines. This suggests that the ability to 'share' resource descriptions in an interoperable format would be beneficial. Competition between the commercial search engines is likely to prevent this from happening. However, subject gateways, particularly those maintained by academics, are more likely to pursue the benefits of sharing/merging their resource descriptions.

6.2 Future Work

The results of an evaluation experiment on the classifier were encouraging (see chapter 4). However the classifier could be greatly improved by:

- Better implementation of the class representatives. The annotation of Dewey proved to be an impossible task for one person to complete. Documents belonging to areas that were well annotated tended to be well classified. In some of these areas the hierarchy was more 'deeply' annotated (i.e. the hierarchy was deeper - there were more subclasses) and classifications were more accurate because the filtering nature of the hierarchy had more chance to take effect. Understandably subject areas where the hierarchy was less deeply annotated worked less well - geography and history (900) were not annotated at all so there was no chance of classifying such pages correctly! Obviously areas that the developer knew more about (computing for example) were probably better annotated than subject areas that the developer knew nothing about. The Dewey for windows CD ROM was used as a reference guide but a lot of additional vocabulary was added where possible. The help of subject experts, possibly librarians, who are familiar with DDC would have been useful here.

- More configurable, flexible implementation of the class representatives. It would be quite simple to write software to automatically extract the (currently hard coded) keywords. It was always the intention to extract the keywords into a more flexible database which would allow them to be edited more easily. The DDC class hierarchy could then be generated in a more dynamic and flexible manner.

- Experimentation with other classification schemes. Although there were good reasons for choosing DDC (see chapter 1), experimentation with other classification schemes might have been interesting/beneficial. Implementation of the kind of flexibility mentioned in the previous point (above) might have made dynamic generation of alternative classification schemes possible. This, however, would require the definition of more than one set of class representatives and so would always have been beyond the scope of this particular project.

- Automatic vocabulary identification. The implementation of a mechanism for identifying commonly coinciding terms within documents assigned to a particular classification area might have been worthwhile. This would enable the system to suggest additional vocabulary for any given class.

In retrospect, it might have been better to focus on particular areas of the classification scheme and define the vocabulary for those areas in detail first rather than attempting to classify everything in one go. Perhaps future developments will refine one branch of the classification hierarchy at a time and improve the overall vocabulary in a more organised incremental manner.

The automatic metadata generator developed for this project demonstrates that automatic metadata generation is possible and has great potential. The system could be improved, however, by:

- Making better use of Dublin Core qualifiers. The decision to develop a new element set (the Wolverhampton Core) was made before DC qualifiers had been properly defined. Recent developments in this area would make pure DC a more realistic and useful option. The specification of the (DC interoperable) WC schema (see appendix K) however, provided the opportunity to investigate the capabilities of RDF schema definition.
- Developing/utilising a more sophisticated abstracting technique. The system currently just grabs the first 25 words from the body of the page or, if present, the contents of the description meta tag. Clearly this could be improved.
- Generating better RDF. The standard was quite new when this system was first developed. With hindsight and a better understanding of RDF it can observed that the generated descriptions are unnecessarily complicated. There is no need to use the two specified container classes for the keywords and classmarks (see chapter 5). The keywords could just be listed within the same *keywords* element not each having its own element within a bag. Similarly the classmarks could be listed within one element not a sequence of elements. The URLs for the schemas (DC and RDF) are out of date as are other syntax details.

6.3 Summary

The contribution of this project has been to show that the automatic classification of web resources, according to a traditional library classification scheme, is possible and has the potential to improve the performance of tools for information resource discovery on the Web. The project also proved that automatic classification can be put to great effect in automatic metadata generation. The metadata generator described in this thesis was presented at the 8th International WWW Conference in Toronto, May 1999, and was among the first RDF applications to be developed.

References

BECKETT, D. (1995) *IAFA Templates in use as Internet Metadata.* http://www.w3.org/conferences/www4/papers/52/. Proceedings of the 4th International World Wide Web Conference, Boston Massachusetts.

BRICKLEY, D. et al. (1998) *Resource Description Framework (RDF) Schema Specification.* http://www.w3.org/TR/WD-rdf-schema

BERNERS-LEE, T (1997) *WWW6 Dev/History*. http://www.w3.org/Talks/9704WWW6-tbl/overview.htm Keynote speech 6th International World Wide Web Conference, Santa Clara.

BURDEN, B. and WALLIS, J. (1995) *Towards a Classification-based Approach to Resource Discovery on the Web*, University of Wolverhampton, http://www.scit.wlv.ac.uk/wwlib/position.html

CHAKRABARTI, S. et al. (1997) *Using taxonomy discriminants and signatures for navigating in text databases*, IBM Almaden Research Centre. Proceedings of the 23rd VLDB conference, Athens, Greece.

CHAN, L. et al. (1996) *Dewey Decimal Classification: A Practical Guide*. Forest Press, ISBN 0-910608-55-5.

CONNOLLY, D. and BOSAK, J. (1998) *Extensible Markup Language (XML).* http://www.w3.org/XML/

DEMPSEY L. and WEIBEL, S. (1996) *The Warwick Metadata Workshop: A Framework for the Deployment of Resource Description* http://www.dlib.org/dlib/july96/07weibel.html, D-Lib Magazine, ISSN 1082-9873

EXCITE INC. (1996) *Information Retrieval technology and Intelligent Concept Extraction (ICE).* http://www.excite.com/ice/tech.html

FAIRTHORNE, R. A. (1961) *The mathematics of classification: Towards Information Retrieval.* Butterworths.

FOSTER, S. and BARRIE, F. (1993) V*eronica-faq*. gopher://veronica.scs.unr.edu

FRAKES, W. B. and BAEZA-YATES, R. (1992) *Information retrieval data structures and algorithms.* Prentice Hall.

GOOD, J. (1958) *Speculations Concerning Information Retrieval*. Research Report PC-78, IBM Research Centre.

HODGSON, J. (2001) *Do HTML Tags Flag Semantic Content?* IEEE Internet Computing, http://computer.org/internet/

KIRRIEMUIR, J. (1996) *Resource Organisation And Discovery in Subject-based Services* (ROADS). http://ukoln.bath.ac.uk/roads/intro.html

KOSTER, M. (1994) *Aliweb*. http://www.nexor.co.uk/public/aliweb/search/doc/form.html

KOSTER, M. (1997) *The Web Robots Page*. http://info.webcrawler.com/mak/projects/robots/

KOWALSKI, G. (1997) *Information Retrieval Systems Theory and Implementation*. Kluwer.

LASSILA, O. and SWICK, R. (1998) *Resource Description Framework (RDF) Model and Syntax Specification*. http://www.w3.org/TR/WD-rdf-syntax

LAWRENCE, S. and GILES, C. L. (1999) *Accessibility of information on the Web*. Nature, Vol. 400 p107 - 109.

LINDER, P. (1992) G*opher-faq*. gopher://ftp.cac.psu.edu/00/internexus/GOPHER.FAQ

LINDOP, L. et al. (1997) *Catching Sites*. PC Magazine, Volume 6, Issue 2, P109-153.

McBRYAN, O. A. (1994) *GENVL and WWWW: Tools for Taming the Web.* http://www.cs.colorado.edu/home/mcbryan/mypapers/www94.ps From proceedings of the First International World Wide Web Conference, Geneva.

MILLER, E. (1998) *The Sixth Dublin Core Metadata Workshop* http://purl.oclc.org/dc/workshops/dc6conference/index.htm

MOENS, M. and DUMORTIER, J. (2000) *Text categorization: the assignment of subject descriptors to magazine articles.* Information Processing & Management, Volume 36, Issue 6, P841-861.

MUSCAT. (1999) *Muscat.* http://www.muscat.com/

NETSACPE. (1999) *The Open Directory Project.* http://dmoz.org/

OCLC FOREST PRESS. (1999) *Dewey Decimal System Home Page.* http://www.oclc.org/oclc/fp/index.htm

OCLC FOREST PRESS. (1996) *Dewey for Windows.* Version 1.0. Asymetrics CD

ODDY, R. N. et al. (1981) *Information Retrieval Research.* Section 4. p35. Butterworths.

PEDERSEN, J and CHANG, W (1999) *Inside Internet Search Engines.* SIGIR'99 conference tutorial notes.

PORTER, M. F. (1980) *An Algorithm for Suffix Stripping.* Program.

RESNICK, P. (1998) *Platform for Internet Content Selection (PICS).* http://www.w3.org/PICS/

RAMBAUGH, J. et al. (1991) *Object-Oriented Modelling and Design.* Prentice Hall International Editions, ISBN 0-13-630054-4.

SALTON, G. (1968) *Automatic Information Organization and Retrieval.* New York: McGraw-Hill

SALTON, G. and McGILL, M. J. (1983) *Introduction to Modern Information Retrieval.* McGraw-Hill.

SALTON, G. WONG, A. and YANG, C. S. (1974) *A Vector Space Model for Automatic Indexing.* http://cs-tr.cs.cornell.edu/Dienst/UI/1.0/Display/ncstrl.cornell/TR74-218

SCHOLZ, A. EFFELSBERG, H. (1999) *The 7th Dublin Core Metadata Workshop.* http://www.ddb.de/partner/dc7conference/index.htm

SODERLAND, S (1997) *Learning to Extract Text-based Information from the World Wide Web,* Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)

SWICK, R. et al. (1998) *Resource Description Framework (RDF).* http://www.w3.org/RDF/

THOMPSON, R. et al. (1997) *Evaluation Dewey Concepts as a Knowledge base for Automatic Subject Assignment.* http://orc.rsch.oclc.org:6109/eval_de.html

VAN RIJSBERGEN, R. C. J. (1981) *Information Retrieval.* Second Edition. Chapter 3, http://www.dcs.glasgow.ac.uk/Keith/Chapter.3/Ch.3.html. Butterworths, ISBN 0-408-10775-8.

WEIBEL, S. et al. (1995) *OCLC/NCSA Metadata Workshop Report* http://purl.oclc.org/dc/workshops/dc1conference/report.htm

WEIBEL, S. IANNELLA, R. and CATHRO, W. (1997) *The 4th Dublin Core Metadata Workshop Report.* http://www.dlib.org/dlib/june97/metadata/06weibel.html. D-Lib Magazine, ISSN 1082-9873

WEIBEL, S. and MILLER, E. (1999) *Dublin Core Metadata.* http://purl.oclc.org/dc/

WEIBEL, S. and MILLER, E. (1997) *Image Description on the Internet: A Summary of the CNI/OCLC Image Metadata Workshop.* http://www.dlib.org/dlib/january97/oclc/01weibel.html. D-Lib Magazine, ISSN 1082-9873

WONG, W. and FU, A. (2000) *Incremental Document Clustering for Web Page Classification,* IEEE Conference on Info. Society in the 21st century: emerging technologies and new challenges (IS2000)

WORLD WIDE WEB CONSORTIUM. (1999) *The World Wide Web Consortium (W3C).* http://www.w3.org

APPENDIX A

Experiment to fix the weight on class representative terms

Adding weight to the keyword objects forming the class representative was found to improve the identification of significant keyword matches in class representatives and documents as discussed in section 3.2.3.4. This appendix documents a short experiment that was carried out to help set the weight that was added to each term across all class representatives.

The URLs used in this experiment are a subset of those used in an early classifier evaluation experiment where librarians were ask to select URLs that pointed at pages in English that were in the UK domain, reasonably wordy and on any topic. From those, ten URLs have been selected for this short experiment covering the widest possible subject areas but each one belonging to very specific domain that the classifier ought to be able to identify.

The following table lists the ten URLs:

1	http://www.notredamecoll.ac.uk	Notre Dame Catholic Sixth Form College
2	http://www.fhgilman.co.uk/html/quarries	Quarries
3	http://www.bham-bot-gdns.demon.co.uk/gardens.html	The Birmingham Botanical Gardens
4	http://www.linguaphone.co.uk	Linguaphone UK
5	http://www.winchester-cathedral.org.uk	The Winchester Cathedral Website
6	http://bubl.ac.uk	BUBL Information Service
7	http://www.steel.org.uk/makstlb.html	The Basic Oxygen Converter
8	http://www.psy.plym.ac.uk	Dr Paul Kenyon's Home Page
9	http://www.animalaid.org.uk	Animal Aid
10	http://www.luminarium.org/medlit	Anthology of Middle English Literature

Class representatives with no weight

The following table lists the result of automatically classifying the above URLs with no weight added from class representative terms. The manual classification assigned by the librarians for each page is also shown (on the right) and a rating of the classifier's performance (using the rating scheme from chapter 4).

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	Classification not identified	0	338
3	630 Agriculture and Related Technologies 580 Plants	3	712
4	420 English 460 Spanish and Portuguese Languages	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	Classification not identified	0	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	820 English and Old English Literatures	4	820
Total		25	

Class representatives with a weight of 2:

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	690 Buildings 380 Commerce, Communications, Transportation	3	338
3	630 Agriculture and Related Technologies 710 Civic and landscape Art	4	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	025.5 Services to users (Operations of libraries, archives, information centres 640 Home Economics and Family Living	3	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	820 English and Old English Literatures 320 Political Science	3	820
Total		31	

Class representatives with a weight of 4, 6 and 8 return the same results as above.
Class representatives with a weight of 10:

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	690 Buildings 380 Commerce, Communications, Transportation	3	338
3	630 Agriculture and Related Technologies 710 Civic and landscape Art	4	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	025.5 Services to users (Operations of libraries, archives, information centres 027 General libraries, archives, information centres	4	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	820 English and Old English Literatures 320 Political Science	3	820
Total		32	

Class representatives with a weight of 12:

URL No.	Classification	Rating	Manual Classification
1	640 Home Economics and Family Living 350 Public Administration and Military Science	2	373
2	690 Buildings 380 Commerce, Communications, Transportation	3	338
3	630 Agriculture and Related Technologies 710 Civic and landscape Art	4	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	025.5 Services to users (Operations of libraries, archives, information centres 307 Communities	3	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	820 English and Old English Literatures 320 Political Science	3	820
Total		29	

A weight of 10 was selected based on these results since the classifications assigned using this weight acquired the highest ratings as shown.

APPENDIX B

Stop List

a
about
above
accordingly
across
after
afterwards
again
against
all
allows
almost
alone
along
already
also
although
always
am
among
amongst
an
and
another
any
anybody
anyhow
anyone
anything
anywhere
apart
appear
appropriate
are
around
as
aside
associated
at
available
away
awfully
b
back
be
became
because
become
becomes
becoming
been
before
beforehand
behind
being
below
beside
besides
best
better
between
beyond
both
brief
but
by
c
came
can
cannot
cant
cause
causes
certain
changes
co
come
consequently
contain
containing
contains
corresponding
could
currently
d
day
described
did
different
do
does
doing
done
down
downwards
during
e
each
eg
eight
either
else
elsewhere
enough
et
etc
even
ever
every
everybody
everyone
everything
everywhere
ex
example
except
f
far
few
fifth
first
five
followed
following
for
former
formerly
forth
four
from
further
furthermore
g
get
gets
given
gives
go
gone
good
got
great
h
had
hardly
has
have
having
he
hence
her
here
hereafter
hereby
herein
hereupon
hers
herself
him
himself
his
hither
how
howbeit
however

i
ie
if
ignored
immediate
in
inasmuch
inc
indeed
indicate
indicated
indicates
inner
insofar
instead
into
inward
is
it
its
itself
j
just
k
keep
kept
know
l
last
latter
latterly
least
less
lest
life
like
little
long
ltd
m
made
make
man
many
may
me
meanwhile
men
might
more
moreover
most
mostly
mr
much
must
my
myself
n
name
namely
near
necessary
neither
never
nevertheless
new
next
nine
no
nobody
none
noone
nor
normally
not
nothing
novel
now
nowhere
o
of
off
often
oh
old
on
once
one
ones
only
onto
or
other
others
otherwise
ought
our
ours
ourselves
out
outside
over
overall
own
p
particular
particularly
people
per
perhaps
placed
please
plus
possible
probably
provides
q
que
quite
r
rather
really
relatively
respectively
right
s
said
same second
secondly
see
seem
seemed
seeming
seems
self
selves
sensible
sent
serious
seven
several
shall
she
should
since
six
so
some
somebody
somehow
someone
something
sometime
sometimes
somewhat
somewhere
specified
specify
specifying
state
still
sub
such
sup
t
take
taken
than
that
the
their
theirs
them
themselves
then
thence
there
thereafter
thereby

therefore
therein
thereupon
these
they
third
this
thorough
thoroughly
those
though
three
through
throughout
thru
thus
time
to
together
too
toward
towards
twice
two
u
under
unless
until
unto
up
upon
us
use
used
useful
uses
using
usually
v
value
various
very
via
viz
vs
w
was
way
we
well
went
were
what
whatever
when
whence
whenever
where
whereafter
whereas
whereby
wherein
whereupon
wherever
whether
which
while
whither
who
whoever
whole
whom
whose
why
will
with
within
without
work
world
would
x
y
year
years
yet
you
your
yours
yourself
yourselves
z
zero
0
1
2
3
4
5
6
7
8
9

APPENDIX C
Experiment to fix the weights associated with certain HTML elements.

Using the structure of HTML to identify terms of particular importance was found to be helpful in the implementation of the original Old ACE classifier (Burden and Wallis, 1996) and has since been endorsed in other research such as that by Soderland (1997) and Hodgson (2001). This appendix documents a short experiment that was carried out to help set the combinations of weights added to terms found within certain tags as described in chapter 3 section 3.2.2.3.

The URLs used in this experiment are the same as those used in appendix A. The tags that were identified as being potentially useful are shown in chapter 3 section 3.2.2.3. A number of different weight combinations were used in deciding upon the weights applied to each tag. The combinations were as follows:

Combination	Tag	Additional Weight
A	<TITLE>	0
	<H1>	0
	<META>	0
	<H2>	0
B	<TITLE>	5
	<H1>	5
	<META>	5
	<H2>	2
C	<TITLE>	10
	<H1>	10
	<META>	10
	<H2>	5
D	<TITLE>	15
	<H1>	15
	<META>	15
	<H2>	10

Combination A - all 0s

The following table lists the result of automatically classifying the same ten URLs introduced in appendix A, with no additional weight added to terms appearing within the specified HTML tags. The manual classification assigned by the librarians for each page is also shown (on the right) and a rating of the classifier's performance (using the rating scheme from chapter 4).

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	Classification not identified	0	338
3	630 Agriculture and Related Technologies	3	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	Classification not identified	0	027
7	Classification not identified	0	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	Classification not identified	0	820
Total		17	

Combination B - 5,5,5,2

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	Classification not identified	0	338
3	630 Agriculture and Related Technologies	3	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	Classification not identified	0	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	Classification not identified	0	820
Total		21	

Combination C - 10,10,10,5

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	690 Buildings 380 Commerce, Communications, Transportation	3	338
3	630 Agriculture and Related Technologies 710 Civic and landscape Art	4	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	025.5 Services to users (Operations of libraries, archives, information centres 027 General libraries, archives, information centres	4	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	820 English and Old English Literatures 320 Political Science	3	820
Total		32	

Combination D - 15,15,15,10

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	690 Buildings 380 Commerce, Communications, Transportation	3	338
3	630 Agriculture and Related Technologies 710 Civic and landscape Art	4	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	025.5 Services to users (Operations of libraries, archives, information centres 640 Home Economics and Family Living	3	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	820 English and Old English Literatures 320 Political Science	3	820
Total		31	

Based on these tests, combination C 10,10,10,5 was chosen.

APPENDIX D
Experiment to fix the threshold for the significance test.

Section 3.2.3.4 of chapter 3 explains how the classifier decides if the keyword matches between a document and a class representative are sufficiently significant for the classifier to proceed to the subclasses of that class or assign the classmark in the case of leaf nodes. This is done by applying a special formula to the total score obtained from the comparison process and testing to see if the result obtained from applying that formula exceeds a certain threshold.

This appendix documents a short experiment carried out to decide upon the value of the threshold. The URLs used in this experiment are the same as those used in appendices A and C.

Threshold of 0

The following table lists the result of automatically classifying the same ten URLs introduced in appendix A, using a threshold of 0. The manual classification assigned by the librarians for each page is also shown (on the right) and a rating of the classifier's performance (using the rating scheme from chapter 4).

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	690 Buildings 380 Commerce, Communications, Transportation	3	338
3	630 Agriculture and Related Technologies 790 Recreational and Performing Arts	3	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	025.5 Services to users (Operations of libraries, archives, information centres 640 Home Economics and Family Living	3	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	820 English and Old English Literatures 320 Political Science	3	820
Total		30	

Threshold of 0.2

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	690 Buildings 380 Commerce, Communications, Transportation	3	338
3	630 Agriculture and Related Technologies 710 Civic and Landscape Art	4	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	025.5 Services to users (Operations of libraries, archives, information centres 640 Home Economics and Family Living	3	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	820 English and Old English Literatures 320 Political Science	3	820
Total		31	

Threshold of 0.5

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	690 Buildings 380 Commerce, Communications, Transportation	3	338
3	630 Agriculture and Related Technologies 710 Civic and landscape Art	4	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	025.5 Services to users (Operations of libraries, archives, information centres 027 General libraries, archives, information centres	4	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	820 English and Old English Literatures 320 Political Science	3	820
Total		32	

Threshold of 0.8

URL No.	Classification	Rating	Manual Classification
1	370 Education 350 Public Administration and Military Science	4	373
2	Classification not identified	0	338
3	630 Agriculture and Related Technologies	3	712
4	190 Modern Western Philosophy 420 English	1	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	025.5 Services to users (Operations of libraries, archives, information centres 027 General libraries, archives, information centres	4	027
7	670 Manufacturing 690 Buildings	4	670
8	370 Education 004.6 Interfacing and communications (Computer science)	2	150
9	590 Animals 630 Agriculture and Related Technologies	4	590
10	820 English and Old English Literatures 320 Political Science	3	820
Total		28	

Threshold of 1

URL No.	Classification	Rating	Manual Classification
1	Classification not identified	0	373
2	Classification not identified	0	338
3	Classification not identified	0	712
4	Classification not identified	0	410
5	260 Christian Social and Ecclesiastical Theology 250 Christian Orders and Local Church	3	250
6	Classification not identified	0	027
7	Classification not identified	0	670
8	Classification not identified	0	150
9	Classification not identified	0	590
10	820 English and Old English Literatures	4	820
Total		7	

Based on these tests, a threshold of 0.5 was chosen.

APPENDIX E

Java Source Code for the Automatic Classification Engine (ACE) classes.

All classes belong to the package jac (Java Automatic Classifier).

E.1 Classmark

```
// This class contains all the methods and instance variables necessary to
// create and maintain a classmark object
package jac;

public class classmark
{
   // The constructor takes two Strings as parameters one representing
   // the textual label and the other the actual classmark
   public classmark(String c, String l)
   {
      classmarkLabel = l;
      cmark = c;
      score = 0;
   }

   // equals returns a boolean set according to whether the classmark
   // matches that of the classmark object passed as a parameter.
   public boolean isequal(classmark cm)
   {
      if (cmark.equals(cm.getClassmark()))
         return true;
      else
         return false;
   }

   // getLabel returns the textual label associated with the classmark
   public String getLabel()
   {
      return classmarkLabel;
   }

   // getClassmark returns the actual classmark
   public String getClassmark()
   {
      return cmark;
   }

   // setScore assigns a score to the classmark - used during the classification
   // process to hold the score relating to a particular class representative
   public void setScore(int n)
   {
      score = n;
   }

   // getScore returns the value of score
   public int getScore()
   {
      return score;
   }

   // isGreater compares the score with the score of the classmark object
   // passed as a parameter.
   public boolean isGreater(classmark cm)
   {
      return score > cm.getScore();
   }

   private String classmarkLabel, cmark;
   private int score;

}
```

E.2 Keyword

```
// This class contains all the methods and instance variables necessary to create
// and maintain a keyword object.

package jac;

public class keyword
{
   // Constructor takes the actual word, its associated score and
   // position (within the document if applicable) as parameters
   public keyword(String w, int s, int p)
   {
      word = w;
      score = s;
      position = p;
   }

   // is_equal returns a boolean depending on whether the word within
   // the keyword object passed as a parameter matches the value of word.
   public boolean is_equal(keyword x)
   {
      if (word.equalsIgnoreCase(x.getKeyword())) return true;
      else return false;
   }

   // getKeyword returns the value of word
   public String getKeyword()
   {
      return word;
   }

   // getScore returns the score
   public int getScore()
   {
      return score;
   }

   // setScore takes an integer as a parameter and uses it to set the score
   public void setScore(int x)
   {
      score = x;
   }

   // incrementScore adds one to the score
   public void incrementScore()
   {
      ++score;
   }

   private String word;
   private int score;
   private int position;

}
```

E.3 Dewey

Dewey is an abstract class inherited by all DDC classes which then use addKeyword, addSubclass and setClassmark to specify their particular properties (example follows).

```
// This abstract class contains all the common methods and instance variables required
// to create and maintain an object containing a DDC class representative

package jac.deweydecimal;

import java.util.*;
import jac.*;

public abstract class dewey
{
   // setClassmark sets up the classmark object which is used to uniquely identify
   // the DDC class
   public void setClassmark(String classm, String ddclabel)
   {
      classMark = new classmark(classm, ddclabel);
   }

   // getClassmark returns the classmark
   public classmark getClassmark()
   {
      return classMark;
   }

   // addKeyword takes a keyword object as a parameter and adds it to the
   // vector of keywords making up the class representative
   public void addKeyword(keyword word)
   {
      keywords.addElement(word);
      totalScore += word.getScore();
   }

   // trimKeywords trims unused elements from the keywords vector
   public void trimKeywords()
   {
      keywords.trimToSize();
   }

   // addSubclass adds the given dewey object to the vector of subclasses
   public void addSubclass(dewey subclass)
   {
      subclasses.addElement(subclass);
   }

   // trimSubclasses trims unused elements from the subclasses vector
   public void trimSubclasses()
   {
      subclasses.trimToSize();
   }

   // setDoneFlag sets the subclassesdone flag - used when adding subclasses
   protected void setDoneFlag()
   {
      subclassesdone=true;
   }

   // getTotalScore returns the value of all keyword scores added together
   public int getTotalScore()
   {
      return totalScore;
   }

   // getTotal returns the length of the class representative - the total number
   // of keywords in the vector.
   public int getTotal()
   {
      return keywords.size();
   }

   // hasMoreKeywords returns a boolean indicating whether all the keywords
```

```
// have been returned from the keywords vector
public boolean hasMoreKeywords()
{
   if (keywords == null)
      return false;
   else
      return marker < keywords.size();
}

// hasMoreSubclasses returns a boolean indicating whether all the subclasses
// have been returned from the subclasses vector
public boolean hasMoreSubclasses()
{
   if (!subclassesdone)
      addSubclasses();
   if (subclasses == null)
      return false;
   else
      return classmarker < subclasses.size();
}


// getNextKeyword returns the next keyword in the vector and increments the marker
public keyword getNextKeyword()
{
   if (marker < keywords.size())
      return (keyword)keywords.elementAt(marker++);
   else
      return null;
}

// getNextSubclass returns the next dewey object in the subclasses vector and
// increments the classmarker.
public dewey getNextSubclass()
{
   if (classmarker < subclasses.size())
      return (dewey)subclasses.elementAt(classmarker++);
   else
      return null;
}

   // noSubclasses sets the subclasses vector to null if there are no subclasses
   // to be added
   protected void noSubclasses()
   {
      subclasses = null;
   }

   // addSubclasses is an abstract method where the appropriate subclasses
   // must be added by DDC classes inheriting this class
   public abstract void addSubclasses();

   protected Vector keywords = new Vector(20,50);
   protected Vector subclasses = new Vector (20,10);
   protected int totalScore=0, marker=0, classmarker=0;
   protected classmark classMark;
   protected boolean subclassesdone=false;
}
```

An example DDC class representing 000 Generalities

This contains a broad class representative for this top level class.

```
package jac.deweydecimal.generalities;

import jac.*;
import jac.deweydecimal.*;
import jac.deweydecimal.generalities.bibliography.*;
import jac.deweydecimal.generalities.library.*;
import jac.deweydecimal.generalities.encyclopedic.*;
import jac.deweydecimal.generalities.museology.*;
import jac.deweydecimal.generalities.media.*;
import jac.deweydecimal.generalities.manuscript.*;

public class generalitiesclass extends dewey
{
    public generalitiesclass()
    {
        setClassmark("000", "Generalities");
        addKeyword(new keyword("knowledge",10,0));
        addKeyword(new keyword("intellect",10,0));
        addKeyword(new keyword("intellectual",10,0));
        addKeyword(new keyword("scholarship",10,0));
        addKeyword(new keyword("humanities",10,0));
        addKeyword(new keyword("statistics",10,0));
        addKeyword(new keyword("statistical",10,0));
        addKeyword(new keyword("honors",10,0));
        addKeyword(new keyword("fellowship",10,0));
        addKeyword(new keyword("fellowships",10,0));
        addKeyword(new keyword("patronage",10,0));
        addKeyword(new keyword("mystery",10,0));
        addKeyword(new keyword("mysteries",10,0));
        addKeyword(new keyword("ufo",10,0));
        addKeyword(new keyword("ufos",10,0));
        addKeyword(new keyword("monster",10,0));
        addKeyword(new keyword("monsters",10,0));
        addKeyword(new keyword("library",10,0));
        addKeyword(new keyword("libraries",10,0));
        addKeyword(new keyword("librarian",10,0));
        addKeyword(new keyword("librarians",10,0));
        addKeyword(new keyword("ddc",10,0));
        addKeyword(new keyword("librarianship",10,0));
        addKeyword(new keyword("ir",10,0));
        addKeyword(new keyword("computer",10,0));
        addKeyword(new keyword("computing",10,0));
        addKeyword(new keyword("cybernet",10,0));
        addKeyword(new keyword("cybernetics",10,0));
        addKeyword(new keyword("smtp",10,0));
        addKeyword(new keyword("computers",10,0));
        addKeyword(new keyword("microcomputers",10,0));
        addKeyword(new keyword("software",10,0));
        addKeyword(new keyword("hypertext",10,0));
        addKeyword(new keyword("database",10,0));
        addKeyword(new keyword("microcomputer",10,0));
        addKeyword(new keyword("cpu",10,0));
        addKeyword(new keyword("pentium",10,0));
        addKeyword(new keyword("intel",10,0));
        addKeyword(new keyword("ibm",10,0));
        addKeyword(new keyword("telnet",10,0));
        addKeyword(new keyword("smtp",10,0));
        addKeyword(new keyword("ethernet",10,0));
        addKeyword(new keyword("ram",10,0));
        addKeyword(new keyword("rom",10,0));
        addKeyword(new keyword("pc",10,0));
        addKeyword(new keyword("mainframe",10,0));
        addKeyword(new keyword("mainframes",10,0));
        addKeyword(new keyword("multiprogramming",10,0));
        addKeyword(new keyword("multitasking",10,0));
        addKeyword(new keyword("multitask",10,0));
        addKeyword(new keyword("real-time",10,0));
        addKeyword(new keyword("realtime",10,0));
        addKeyword(new keyword("multiprocessing",10,0));
        addKeyword(new keyword("vdu",10,0));
        addKeyword(new keyword("microform",10,0));
```

```
addKeyword(new keyword("multiprocess",10,0));
addKeyword(new keyword("multiprocessor",10,0));
addKeyword(new keyword("videotex",10,0));
addKeyword(new keyword("api",10,0));
addKeyword(new keyword("comms",10,0));
addKeyword(new keyword("baseband",10,0));
addKeyword(new keyword("broadband",10,0));
addKeyword(new keyword("modem",10,0));
addKeyword(new keyword("modems",10,0));
addKeyword(new keyword("multiplex",10,0));
addKeyword(new keyword("multiplexing",10,0));
addKeyword(new keyword("wan",10,0));
addKeyword(new keyword("wans",10,0));
addKeyword(new keyword("dfd",10,0));
addKeyword(new keyword("elh",10,0));
addKeyword(new keyword("ssadm",10,0));
addKeyword(new keyword("jsp",10,0));
addKeyword(new keyword("pseudocode",10,0));
addKeyword(new keyword("dataflow",10,0));
addKeyword(new keyword("data-flow",10,0));
addKeyword(new keyword("omt",10,0));
addKeyword(new keyword("supercomputer",10,0));
addKeyword(new keyword("supercomputers",10,0));
addKeyword(new keyword("supercomputing",10,0));
addKeyword(new keyword("minicomputer",10,0));
addKeyword(new keyword("minicomputers",10,0));
addKeyword(new keyword("minicomputing",10,0));
addKeyword(new keyword("laptop",10,0));
addKeyword(new keyword("palmtop",10,0));
addKeyword(new keyword("processor",10,0));
addKeyword(new keyword("processors",10,0));
addKeyword(new keyword("cisc",10,0));
addKeyword(new keyword("risc",10,0));
addKeyword(new keyword("human-computer",10,0));
addKeyword(new keyword("microcomputing",10,0));
addKeyword(new keyword("programming",10,0));
addKeyword(new keyword("algorithm",10,0));
addKeyword(new keyword("algorithms",10,0));
addKeyword(new keyword("cgi",10,0));
addKeyword(new keyword("java",10,0));
addKeyword(new keyword("fortran",10,0));
addKeyword(new keyword("c++",10,0));
addKeyword(new keyword("cobol",10,0));
addKeyword(new keyword("vb",10,0));
addKeyword(new keyword("modula2",10,0));
addKeyword(new keyword("applet",10,0));
addKeyword(new keyword("applets",10,0));
addKeyword(new keyword("software",10,0));
addKeyword(new keyword("yahoo",10,0));
addKeyword(new keyword("flat-file",10,0));
addKeyword(new keyword("infoseek",10,0));
addKeyword(new keyword("lycos",10,0));
addKeyword(new keyword("hotbot",10,0));
addKeyword(new keyword("webcrawler",10,0));
addKeyword(new keyword("altavista",10,0));
addKeyword(new keyword("object-oriented",10,0));
addKeyword(new keyword("awk",10,0));
addKeyword(new keyword("debugging",10,0));
addKeyword(new keyword("debugger",10,0));
addKeyword(new keyword("visual-basic",10,0));
addKeyword(new keyword("motif",10,0));
addKeyword(new keyword("spreadsheet",10,0));
addKeyword(new keyword("spreadsheets",10,0));
addKeyword(new keyword("unix",10,0));
addKeyword(new keyword("dos",10,0));
addKeyword(new keyword("microprogram",10,0));
addKeyword(new keyword("microprogramming",10,0));
addKeyword(new keyword("microcode",10,0));
addKeyword(new keyword("ai",10,0));
addKeyword(new keyword("virtual-reality",10,0));
addKeyword(new keyword("vr",10,0));
addKeyword(new keyword("knowledge-based",10,0));
addKeyword(new keyword("ocr",10,0));
addKeyword(new keyword("knowledge-base",10,0));
addKeyword(new keyword("knowledgebase",10,0));
addKeyword(new keyword("knowledgebased",10,0));
addKeyword(new keyword("expert-system",10,0));
```

```
        addKeyword(new keyword("photoshop",10,0));
        addKeyword(new keyword("bitmap",10,0));
        addKeyword(new keyword("xv",10,0));
        addKeyword(new keyword("bmp",10,0));
        addKeyword(new keyword("pixel",10,0));
        addKeyword(new keyword("pixels",10,0));
        addKeyword(new keyword("bibliography",10,0));
        addKeyword(new keyword("bibliographies",10,0));
        addKeyword(new keyword("bibliographic",10,0));
        addKeyword(new keyword("encyclopedic",10,0));
        addKeyword(new keyword("encyclopedia",10,0));
        addKeyword(new keyword("encyclopedias",10,0));
        addKeyword(new keyword("museology",10,0));
        addKeyword(new keyword("museum",10,0));
        addKeyword(new keyword("museums",10,0));
        addKeyword(new keyword("documentary",10,0));
        addKeyword(new keyword("journalism",10,0));
        addKeyword(new keyword("journalist",10,0));
        addKeyword(new keyword("documentaries",10,0));
        addKeyword(new keyword("newspaper",10,0));
        addKeyword(new keyword("newspapers",10,0));
        addKeyword(new keyword("tabloid",10,0));
        addKeyword(new keyword("tabloids",10,0));
        addKeyword(new keyword("newsletter",10,0));
        addKeyword(new keyword("newsletters",10,0));
        addKeyword(new keyword("newsreel",10,0));
        addKeyword(new keyword("newsreels",10,0));
        addKeyword(new keyword("broadcast",10,0));
        addKeyword(new keyword("broadcasting",10,0));
        addKeyword(new keyword("broadcasted",10,0));
        addKeyword(new keyword("bbc",10,0));
        addKeyword(new keyword("radio",10,0));
        addKeyword(new keyword("wireless",10,0));
        addKeyword(new keyword("television",10,0));
        addKeyword(new keyword("tv",10,0));
        addKeyword(new keyword("reporters",10,0));
        addKeyword(new keyword("photojournalism",10,0));
        addKeyword(new keyword("publishers",10,0));
        addKeyword(new keyword("manuscript",10,0));
        addKeyword(new keyword("manuscripts",10,0));
        addKeyword(new keyword("incunabula",10,0));
        trimKeywords();
    }

    public void addSubclasses()
    {
        addSubclass(new systems());
        addSubclass(new computerscience());
        addSubclass(new programming());
        addSubclass(new computermethods());
        addSubclass(new bibliographyclass());
        addSubclass(new libraryclass());
        addSubclass(new encyclopedicclass());
        addSubclass(new museologyclass());
        addSubclass(new mediaclass());
        addSubclass(new manuscriptclass());
        trimSubclasses();
        setDoneFlag();
    }
}
```

E.4 Document

```
// This class provides all the methods and instance variables needed to create and
// maintain a document object.

package jac;

import java.io.*;
import java.util.*;
import java.net.*;

public class document
{

   // Constructor takes an open DataInputStream and an integer representing the
   // accession number as parameters and indexes the file to produce a vector
   // of keywords
   public FileDocument (DataInputStream dis, int n) throws IOException
   {

      accession = n;
      docfile = new BufferedReader(new InputStreamReader(fis));

      doindexing();
      docfile.close();
   }


   public void resetKeywords()
   {
      marker = 0;
   }



   public void resetClassmarks()
   {
      classmarkmarker =0;
   }



   public int getAccession()
   {
      return accession;
   }



   public int getTotalScore()
   {
      return totalScore;
   }



   public int getTotal()
   {
      keywords.trimToSize();
      return keywords.size();
   }

   public int getWordCount()
   {
      return wordCount;
   }


   public boolean hasMoreKeywords()
   {
      return marker < keywords.size();
   }
```

```
public boolean hasMoreClassmarks()
{
   return classmarkmarker < classmarks.size();
}


public keyword getNextKeyword()
{
   if (marker < keywords.size())
      return (keyword)keywords.elementAt(marker++);
   else
      return null;
}


public classmark getNextClassmark()
{
   classmark cm;

   if (classmarkmarker < classmarks.size())
      return (classmark)classmarks.elementAt(classmarkmarker++);
   else
      return null;
}


public void addClassmark(classmark classMark)
{
   classmarks.addElement(classMark);
}




private void doindexing() throws IOException
{
      String line, word;                  // for each line and each word
      StringTokenizer words;              // for tokenizing the line
      Stop stops = new Stop();            // stop word detector

      line = docfile.readLine();          // read the first line

      while (line != null)                // while not end of file
      {
         line = noHTML(line);             // remove HTML from the line read
         words = new StringTokenizer( line, "£\'#<>_+*&^%\\/@~|.,?;:!\"(){}[]-=
\t\n\r" );
                                          // tokenize the line
         while(words.hasMoreTokens())
         {
            word = words.nextToken(); // get each word
            if (!stops.inStop(word))  // if it's not a stop word
               addWord(word);         // add it
         }
         line = docfile.readLine();   // read the next line
      }
      keywords.trimToSize();
   }


   private String noHTML(String s)
   {
      int charCount = 0;
      String newString = " ";

      while (charCount < s.length())
      {
         if (s.charAt(charCount) == '<')
            tag = true;
         else if (s.charAt(charCount) == '>')
         {
            tag = false;
```

```
            newString = newString + processTag(tagString);
            tagString = " ";
         }
         else if (tag)
            tagString = tagString + s.charAt(charCount);
         else
            newString = newString + s.charAt(charCount);
         charCount++;
      }
      return removeSpecialChars(newString);
   }

   private String processTag(String tagStr)
   {
      StringTokenizer attributes;
      String returnString = " ", attribute;

      attributes = new StringTokenizer(tagStr, " \"_-+=:;\'.,\\!$%^&*()?@{}[]\n\t\r");
      if (attributes.hasMoreTokens())
      {
         attribute = attributes.nextToken();
         if (attribute.equalsIgnoreCase("TITLE"))
            returnString = returnString + " jacmarker1 ";
         else if (attribute.equalsIgnoreCase("/TITLE"))
            returnString = returnString + " jacmarkerend ";
         else if (attribute.equalsIgnoreCase("H1"))
            returnString = returnString + " jacmarkerh1 ";
         else if (attribute.equalsIgnoreCase("/H1"))
            returnString = returnString + " jacmarkerh1end ";
         else if (attribute.equalsIgnoreCase("H2"))
            returnString = returnString + " jacmarkerh2 ";
         else if (attribute.equalsIgnoreCase("/H2"))
            returnString = returnString + " jacmarker2end ";
         else if ((attribute.equalsIgnoreCase("meta")) &&
(attributes.hasMoreTokens()))
         {
            attribute = attributes.nextToken();
            if ((attribute.equalsIgnoreCase("name")) && (attributes.hasMoreTokens()))
            {
               attribute = attributes.nextToken();
               if (((attribute.equalsIgnoreCase("description")) ||
(attribute.equalsIgnoreCase("keywords"))) && (attributes.hasMoreTokens()))
               {
                  attribute = attributes.nextToken();
                  if ((attribute.equalsIgnoreCase("content")) &&
(attributes.hasMoreTokens()))
                  {
                     returnString = returnString + " jacmarkermeta ";
                     while (attributes.hasMoreTokens())
                        returnString = returnString + " " + attributes.nextToken() + "
";
                     returnString = returnString + " jacmarkermetaend ";
                  }
               }
            }
         }
      }
      return returnString;
   }

   private String removeSpecialChars(String str)
   {
      String returnString = " ";

      // tokenize on &
      StringTokenizer tokens = new StringTokenizer(str, "&");
      while (tokens.hasMoreTokens()) // check each token for special chars
         returnString = returnString + charChecker(tokens.nextToken());
      return returnString;
   }

   private String charChecker(String str)
   {
      String returnString = " ";

      // tokenize on ;
      StringTokenizer tokens = new StringTokenizer(str, ";");
```

```
    if (tokens.hasMoreTokens())
    {  // if there is a token see if it is a special char
       if (specialChar(tokens.nextToken()))
       {  // if so send back the remaining string
          while (tokens.hasMoreTokens())
             returnString = returnString + tokens.nextToken();
       }   // else send back the string as was
       else
          returnString = str;
    }
    else
       returnString = str;
    return returnString;
}


private boolean specialChar(String entity)
{
    String strNum;
    int num, index = 0;
    boolean found = false;

    if (entity.length() < 7)
    {

       // see if the entity is found in the alphaCharEntities array
       while((index < alphaCharEntities.length) && (!found))
       {
          if (entity.equalsIgnoreCase(alphaCharEntities[index]))
             found = true;
          index++;
       }

       // if its not found in the array ...
       if (!found)
       {   // and has more that 2 characters ...
          if (entity.length() > 2)
          {
             // test for numerical character entity
             if (entity.charAt(0) == '#')
             {
                try
                {
                   strNum = entity.substring(1, (entity.length() - 1));

                   num = Integer.parseInt( strNum );

                   if ( ( num == 38 ) || (num == 160) ||
                        ( num >= 48 && num <= 57 ) ||
                        ( num >= 65 && num <= 90 ) ||
                        ( num >= 97 && num <= 122 ) ||
                        ( num >= 192 && num <= 214 ) ||
                        ( num >= 216 && num <= 246 ) ||
                        ( num >= 248 && num <= 255 ) )
                     found = true;
                }
                catch (NumberFormatException nfe){}
             }
          }
       }
    }
    return found;
}


private void addWord(String word)
{
    if (word.equals("jacmarker1"))
    {
       if (!past_title)
       {
          score += 9;
          past_title = true;
          title = true;
       }
    }
    else if (word.equals("jacmarkerend"))
```

```
{
   if (title)
   {
      score -= 9;
      title = false;
   }
}
else if (word.equals("jacmarkerh1"))
{
   if (!heading)
   {
      score += 9;
      heading = true;
   }
}
else if (word.equals("jacmarkerh1end"))
{
   if (heading)
   {
      score -= 9;
      heading = false;
   }
}
else if (word.equals("jacmarkerh2"))
{
   if (!heading)
   {
      score += 4;
      heading = true;
   }
}
else if (word.equals("jacmarkerh2end"))
{
   if (heading)
   {
      score -= 4;
      heading = false;
   }
}
else if (word.equals("jacmarkermeta"))
{
      if (!meta)
      {
         score += 9;
         meta = true;
      }
   }
   else if (word.equals("jacmarkermetaend"))
   {
      if (meta)
      {
         score -= 9;
         meta = false;
      }
   }
   else
   {
      wordnumber++;
      keywords.addElement(new keyword(word, score, wordnumber));
      totalScore += score;
      if (!title && !meta)
         wordCount++;
   }
}
private BufferedReader docfile;
   private Vector keywords = new Vector(20,50);
   private Vector classmarks = new Vector(5,5);
   private int accession, totalScore=0, classmarkmarker=0, marker=0, wordCount=0;
   private int wordnumber=0, score=1;
   private String tagString = " ";
   private boolean past_title=false, title = false, heading=false, meta=false;
   private static boolean tag = false;
   private String[] alphaCharEntities = { "Agrave", "Aacute", "Acirc",  "Atilde",
"Amp", "Auml",   "Aring",  "AElig",  "Ccedil", "lt", "Egrave", "Eacute", "Ecirc",
"Euml", "gt", "Igrave", "Iacute", "Icirc",  "Iuml", "nbsp", "ETH", "Ntilde", "Ograve",
"Oacute", "Ocirc",  "Otilde", "Ouml",   "Oslash", "Ugrave", "Uacute", "Ucirc", "Uuml",
"Yacute", "THORN",  "szlig"};
```

```
}
```

E.5 Classify

```
// This class takes a document object (as a parameter on the constructor) and
// proceeds to classify it by comparing it with each branch of the DDC hierarchy.
package jac;

import java.io.*;
import jac.deweydecimal.*;
import jac.deweydecimal.generalities.*;
import jac.deweydecimal.philosophy.*;
import jac.deweydecimal.religion.*;
import jac.deweydecimal.socialsciences.*;
import jac.deweydecimal.language.*;
import jac.deweydecimal.naturalsciences.*;
import jac.deweydecimal.technology.*;
import jac.deweydecimal.arts.*;
import jac.deweydecimal.literature.*;

// Constructor takes a document as a parameter and calls the proceed method for
// each branch of the hierarchy.
public class classify
{

   public classify(document d)
   {
      doc = d;
      proceed(new generalitiesclass());
      proceed(new philosophyclass());
      proceed(new religionclass());
      proceed(new socialsciencesclass());
      proceed(new languageclass());
      proceed(new naturalsciencesclass());
      proceed(new technologyclass());
      proceed(new artsclass());
      proceed(new literatureclass());
/*      proceed(new geoghistory());  not implemented */
   }

   // getClassmarks returns the classmarks from the document
   public String getClassmarks()
   {
      return doc.getClassmarks();
   }


// proceed takes a dewey object as a parameter and scores it against
// (compares it with) the document.  If the score is significant, the
// proceed method is then called recursively for each of any subclasses.
private void proceed(dewey ddc)
{
   classmark cm;
   int totalscore;

   totalscore = score(ddc);
   if (significant(totalscore, ddc.getTotal(), doc.getTotal()))
   {
      if (!ddc.hasMoreSubclasses())
      {
         cm = ddc.getClassmark();
         cm.setScore(totalscore);
         doc.addClassmark(cm);
      }
      else
      {
         while (ddc.hasMoreSubclasses())
            proceed(ddc.getNextSubclass());
      }
   }
}


// significant takes the total score associated with a document/class
// comparison, the length of the class and the length of the document
// and calculates the Dice Coefficient.
private boolean significant(int totalscore, int deweylength, int doclength)
```

```
{
   float totalLength = deweylength + doclength;

   if ((2 * (totalscore / totalLength)) > 0.5)
      return true;
   else
      return false;
}

// score takes a dewey object as a parameter and compares each word in the
// dewey keyword vector with each word in the document keyword vector resulting
// in a total score
public int score(dewey ddc)
{
   int thescore=0,doccount=0, deweycount=0;
   keyword docword, deweyword;

      while (ddc.hasMoreKeywords())
      {
         deweyword = ddc.getNextKeyword();
         while (doc.hasMoreKeywords())
         {
            docword = doc.getNextKeyword();
            if (deweyword.is_equal(docword))
               thescore = thescore + deweyword.getScore() + docword.getScore();

         }
         doc.resetKeywords();
      }
      return thescore;
   }

   private document doc;
}
```

E.6 ACE

```
// This class co-ordinates the classification process by opening the document,
// generating a document object and passing that document object as a parameter to
// an instance of the classify object
import jac.*;
import java.io.*;
import java.util.*;
import java.net.*;


public class ace
{
   public static void main (String[] args)
   {
      DataInputStream docfile;
      URL url;
      HttpURLConnection uc;
      long lastModLong = 0;


      if (args.length == 0)
         System.out.println("Usage: java ace <filename>\nUsage: java ace -url <URL>");
      else
      {
         if (args[0].equals("-url"))
         {
            if (args.length < 2)
            {
               System.out.println("Usage: java ace <filename>\nUsage: java ace -url
<URL>");
               System.exit(1);
            }
            else
            {
               try
               {
                  url = new URL(args[1]);
                  docfile = new DataInputStream(url.openStream());
                  Doc = new document(docfile,0);
                  classification = new classify(Doc);
                  System.out.print(classification.getClassmarks());
                  docfile.close();
               }
               catch (MalformedURLException mu)
               {
                  System.out.println ("Sorry cannot find URL: " + args[1]);
                  System.exit(1);
               }
               catch (IOException e)
               {
                  System.out.println ("Sorry cannot connect to URL: " + args[1]);
                  System.exit(1);
               }
         }
      }
      else
      {
         try
         {
            docfile = new DataInputStream(new FileInputStream(args[0]));
            Doc = new document(docfile,0);
            classification = new classify(Doc);
            System.out.print(classification.getClassmarks());
            docfile.close();
         }
         catch (IOException e)
         {
            System.out.println ("Error reading file " + args[0]);
            System.exit(1);
         }
      }
   }
}
```

```
    private static document Doc = null;
    private static classify classification;

}
```

E.7 Stop

```
// This class creates a stop word object.  Stop words are read from a text file -
stop.txt - into an internal
// data structure which can then be used to check if new words encountered in
documents are stop words.
// Charlotte Jenkins February 2000

package jac;

import java.io.*;
import java.util.*;

public class Stop
{
   public Stop()
   {
      BufferedReader inFile;              // BufferedReader variable for reading the
stop file
      String fileName, line;              // String variables for storing the filename
and each line read
      StringTokenizer tokens;             // StringTokenizer variable for breaking each
line into words

      try
      {
         fileName="stop.txt";
         inFile = new BufferedReader(new FileReader(fileName));
         line = inFile.readLine();        // Reads a line from the file into line
         while (line != null)             // While line is not null
         {
            tokens = new StringTokenizer(line, " \n\r");   // tokenize the line into
words
            while (tokens.hasMoreTokens())                 // while there are more words
               stopwords.addElement(tokens.nextToken());   // add each word to the
vector
            line = inFile.readLine();                      // read the next line from the
file
         }                                                 // no more lines to read
         inFile.close();                                   // close the file
         stopwords.trimToSize();                           // trim the vector
      }
      catch (IOException e)                                // catch IO exceptions
      {
         System.out.println("Problem reading stop word file\n" + e);
      }
   }

   //-----------------------------------------------------------------------------
-------------------------------

   // inStop returns a boolean value depending on whether the word passed as a
parameter is found in the list

   public boolean inStop (String word)
   {
      int index=0;
      boolean found = false;

      while ((index < stopwords.size()) && (!found))
         if (((String)stopwords.elementAt(index++)).equalsIgnoreCase(word))
            found = true;
      return found;
   }

   //-----------------------------------------------------------------------------
-------------------------------

   // getStopWords returns one String containing all the stop words

   public String getStopWords()
   {
     int index=0;
     String stops=null;
```

```
        while (index < stopwords.size())
        {
           if (stops == null)
              stops = (String)stopwords.elementAt(index++) + " ";
           else
              stops = stops + (String)stopwords.elementAt(index++) + " ";

        }
        return stops;
     }


     private Vector stopwords = new Vector(30); // Vector where stop words are stored
}
```

APPENDIX F

Source code of classes required for client/server application.

There follows the additional classes required to implement the client/server application referred to in section 3.3.2.

F.1 ThreadedAceServer

The server listens to port 8189 and passes any input to ThreadedAceHandler (see next subsection)

```
import java.io.*;
import java.net.*;
import java.util.*;

public class ThreadedAceServer
{
    public static void main(String[] args)
    {
          int i=1;
          try
          {
                ServerSocket s = new ServerSocket(8189);
                for (;;)
                {
                     Socket incoming = s.accept();
                     System.out.println("Spawning " + i + "\n");
                     new ThreadedAceHandler(incoming, i).start();
                     i++;
                }
          }
          catch (Exception e)
          {
                System.out.println(e);
          }
  }
}
```

F.2 ThreadedAceHandler

This class basically replaces the original ace. It inherits the Thread object and as such every instance of this object acts as a thread so that multiple classifiers can be run simultaneously if more than one client happens to access the server at once.

```
import jac.*;
import java.io.*;
import java.util.*;
import java.net.*;

public class ThreadedAceHandler extends Thread
{
        public ThreadedAceHandler(Socket i, int c)
        {
        incoming = i;
        counter = c;
}

public void run()
{

        try
        {
                in = new DataInputStream(incoming.getInputStream());
                out = new PrintStream(incoming.getOutputStream());
                out.println ("Connection established");
                out.println ("Classifying...");
                doclassification();
                incoming.close();
        }
        catch (Exception e)
        {
                System.out.println(e);
        }
}

public void doclassification()
{
        document Doc = null;
        classify classification;
        URL url;
        DataInputStream docfile;
        String nexturl = "<no URL>";

        try
        {
                nexturl = in.readLine();
                url = new URL(nexturl);
                docfile = new DataInputStream(url.openStream());
                Doc = new document(docfile,0);
                classification = new classify(Doc);
                out.print(classification.getClassmarks());

        }
        catch (MalformedURLException mu)
        {
                out.println ("Sorry cannot find URL: " + nexturl);
        }
        catch (IOException e)
        {
                out.println ("Sorry cannot connect to URL: " + nexturl);
                }
        }


        private Socket incoming;
        private int counter;
        private DataInputStream in;
        private PrintStream out;

}
```

APPENDIX G

Program to randomly select test set from 20000 classifications.

This program reads the results file for the 20000 classifications and randomly selects a subset of 200 as described in section 4.2 of chapter 4.

```
// Need to provide maximum = line count on selectedresults.txt
import java.io.*;
import java.util.*;

public class SelectFinalResults
{
   private static int[] randomNumbers;
   private static int selectedCount = 0;

   public static void main(String[] args)
   {

      String line;
      int lineCount = 0, nextNo;
      Random rng = new Random();
      BufferedReader in;

      randomNumbers = new int[200];
      int maximum = Integer.valueOf(args[0]).intValue();

      try
      {

         PrintWriter finalResults = new PrintWriter(new
                                                FileWriter("finalresults.txt"));

         while (selectedCount < 200)         // while less than 200 have been selected
         {
            nextNo = rng.nextInt(maximum);   // generate a random number
            while (inList(nextNo))           // get another if it's already been used
               nextNo = rng.nextInt(maximum);

            in = new BufferedReader(new FileReader("selectedresults.txt"));
            lineCount = 0;
            line = in.readLine();
            while ((line != null) && (lineCount < nextNo))
            {
               line = in.readLine();
               lineCount++;
            }
            if (line != null)
               finalResults.println(line);
            in.close();
         }
         System.out.println ("The number of selections made was " + selectedCount);
         finalResults.close();
      }
      catch (IOException e)
      {
         System.out.println("Error: " + e);
      }
   }

   private static boolean inList(int randomNumber)
   {
      int index = 0;
      while (index < selectedCount)
      {
         if (randomNumbers[index++] == randomNumber)
            return true;
      }
      randomNumbers[selectedCount++] = randomNumber;
      return false;
   }
}
```

APPENDIX H

Instructions given to Librarians.

The following instructions were posted on the web for the librarians to follow:

This page contains instructions for the evaluation of an automatic classifier that has been developed by Charlotte Jenkins as part of her PhD work. These instructions are to be followed by librarians from the University of Hull Scarborough Campus who have kindly volunteered to assist in this evaluation experiment.

Each librarian must follow these instructions independently of the others.

On the web page with the URL http://www.scit.wlv.ac.uk/~ex1253/experiment/, you will find a table comprising 200 rows. Each of these rows represents a web page that has been automatically classified according to Dewey Decimal Classification.

In order to carry out this evaluation experiment it will be necessary to have the on-line version of this table in front of you in addition to a printed, paper version. The on-line version will be used to follow links to copies of the automatically classified pages and the paper version will be used for indicating a rating for each automatic classification and for assigning a manual classification for each page.

Each row of the table has the following columns:

- **Number:** 1 - 200
- **Local:** this is a hyperlink to a local copy of the web page. This is the version of the web page that has been automatically classified by the classifier.
- **Remote:** this is a hyperlink to the remote (live, on-line) version of the page which may have changed considerably (or even disappeared) since the automatic classification of the local copy. This link is there in case the local copy is incomprehensible to the human classifier (due to a lack of images or other externally referenced files). The remote version may be viewed by the librarians to assist in the evaluation process but the rating must ultimately be based on the classification of the local copy.
- **Classmarks:** these are the automatically assigned DDC classmarks. Many of the pages have been assigned two classmarks, in these instances both classmarks must be rated independently.
- **Rating:** this column is split into four sub-columns numbered 1 - 4. Each automatic classification must be rated by entering a tick in one of these four columns. The rating scheme is defined below.
- **Manual classification:** each librarian must manually assign a broad classification for each page in this final column.

Rating Scheme

1. tick under this column if this classification is completely inappropriate for this page.
2. tick under this column if, although not completely inappropriate, you would be surprised to see this page classified under this classmark in an on-line web directory.
3. tick under this column if, although not entirely accurate, you do not feel it would be misleading to see this page classified under this classmark in an on-line web directory.
4. tick under this column if this classification is accurate. This is where you would expect to see this page classified in an on-line web directory.

APPENDIX I
Independent Results from Librarians.

The following pages show the independent results from librarians 1, 2 and 3.

Classifier Results

Librarian 1

No.	Local	Remote	Classmarks	Rating 1	2	3	4	Manual classification
1	local copy	remote version	370 Education				✓	378
						✓		
2	local copy	remote version	640 Home Economics and Family Living	✓				696.1
3	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004.6
			005.7 Data in computer systems			✓		
4	local copy	remote version	640 Home Economics and Family Living				✓	647.94423 42
			790 Recreational and Performing Arts		✓			
5	local copy	remote version	320 Political Science				✓	320
			190 Modern Western Philosophy			✓		
6	local copy	remote version	070.1 Documentary media, educational media, news media			✓		302.2
			340 Law	✓				
7	local copy	remote version	690 Buildings				✓	690
			640 Home Economics and Family Living		✓			
8	local copy	remote version	640 Home Economics and Family Living	✓				338.7
			380 Commerce, Communications, Transportation			✓		
9	local copy	remote version	380 Commerce, Communications, Transportation	✓				378.11
10	local copy	remote version	370 Education				✓	372
			304 Social Behaviour		✓			
11	local copy	remote version	510 Mathematics			✓		507.11
			150 Psychology	✓				
12	local copy	remote version	330 Economics		✓			650.14
			650 Management and Auxiliary Services				✓	
13	local copy	remote version	708 Galleries, Museums, Private Collections				✓	708
			069 Museology (Museum science)			✓		
14	local copy	remote version	610 Medical Sciences, Medicine				✓	610.730711
			370 Education			✓		
15	local copy	remote version	520 Astronomy and Allied Sciences	✓				338.7
16	local copy	remote version	307 Communities				✓	307
			305 Social Groups			✓		

				1	2	3	4	
17	local copy	remote version	003.5 Theory of communication and control (Computer Systems)		✓	✓		338.7072
18	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004.6
			680 Manufacture for Specific uses	✓				
19	local copy	remote version	370 Education		✓			021
			021 Relationships of libraries, archives, information centres				✓	
20	local copy	remote version	790 Recreational and Performing Arts				✓	790
			005.3 Programs (Computer programs)			✓		
21	local copy	remote version	370 Education		✓			021
			021 Relationships of libraries, archives, information centres				✓	
22	local copy	remote version	380 Commerce, Communications, Transportation			✓		331.8811365
23	local copy	remote version	650 Management and Auxiliary Services	✓				020.711
			027 General libraries, archives, information centres				✓	
24	local copy	remote version	590 Animals	✓				133.3337
			690 Buildings		✓			
25	local copy	remote version	340 Law	✓				338.4791099646
26	local copy	remote version	670 Manufacturing	✓				700.285
			005.7 Data in computer systems			✓		
27	local copy	remote version	610 Medical Sciences, Medicine	✓				570.79
			370 Education			✓		
28	local copy	remote version	005.7 Data in computer systems			✓		570.79
			005.1 Programming (Computer programming)	✓				
29	local copy	remote version	070.1 Documentary media, educational media, news media			✓		070.4
			070.4 Journalism				✓	
30	local copy	remote version	027 General libraries, archives, information centres				✓	027
			690 Buildings	✓				
31	local copy	remote version	340 Law			✓		658.88
32	local copy	remote version	370 Education				✓	372.241
			650 Management and Auxiliary Services	✓				

33	local copy	remote version	070.1 Documentary media, educational media, news media		✓			791.437
			350 Public Administration and Military Science	✓				
34	local copy	remote version	820 English and Old English Literatures		✓			381.45002
			790 Recreational and Performing Arts		✓			
35	local copy	remote version	027 General libraries, archives, information centres	✓				378.161
			022 Administration of the physical plant (Library and information sciences)	✓				
36	local copy	remote version	790 Recreational and Performing Arts	✓				352.669
			005.8 Data security	✓				
37	local copy	remote version	670 Manufacturing	✓				150.92
			380 Commerce, Communications, Transportation	✓				
38	local copy	remote version	790 Recreational and Performing Arts		✓			338.4791
			004.6 Interfacing and communications (Computer science)	✓				
39	local copy	remote version	780 Music				✓	784.2
			005.1 Programming (Computer programming)	✓				
40	local copy	remote version	027 General libraries, archives, information centres		✓			335.4092
			420 English	✓				
41	local copy	remote version	790 Recreational and Performing Arts				✓	793.932
			780 Music	✓				
42	local copy	remote version	004.1 General works on specific types of computers	✓				331.128
			005.4 Systems programming and programs (Computer programs)	✓				
43	local copy	remote version	790 Recreational and Performing Arts			✓		780
			780 Music				✓	
44	local copy	remote version	790 Recreational and Performing Arts				✓	796.8152
45	local copy	remote version	307 Communities				✓	307
			305 Social Groups			✓		
46	local copy	remote version	370 Education			✓		570.79
			350 Public Administration and Military Science	✓				
47	local copy	remote version	510 Mathematics				✓	510.92
			370 Education			✓		

48	local copy	remote version	025.5 Services to users (Operations of libraries, archives, information centres	✓				004.07114221
			027 General libraries, archives, information centres	✓				
49	local copy	remote version	070.1 Documentary media, educational media, news media	✓				914.604
50	local copy	remote version	370 Education				✓	371.67
			027 General libraries, archives, information centres	✓				
51	local copy	remote version	027 General libraries, archives, information centres				✓	027
			610 Medical Sciences, Medicine		✓			
52	local copy	remote version	340 Law			✓		323.09951
			320 Political Science				✓	
53	local copy	remote version	025.5 Services to users (Operations of libraries, archives, information centres				✓	025.5
			380 Commerce, Communications, Transportation		✓			
54	local copy	remote version	004.1 General works on specific types of computers			✓		027
			027 General libraries, archives, information centres				✓	
55	local copy	remote version	340 Law	✓				914.28804
			630 Agriculture and Related Technologies	✓				
56	local copy	remote version	370 Education			✓		004.6
			004.6 Interfacing and communications (Computer science)				✓	
57	local copy	remote version	750 Painting and Paintings	✓				914.604
			740 Drawing and Decorative Arts	✓				
58	local copy	remote version	370 Education			✓		530.071142496
			530 Physics				✓	
59	local copy	remote version	340 Law				✓	340.07114134
			370 Education			✓		
60	local copy	remote version	004.6 Interfacing and communications (Computer science)	✓				368.0065
			005.3 Programs (Computer programs)	✓				
61	local copy	remote version	590 Animals	✓				025.04
62	local copy	remote version	820 English and Old English Literatures	✓				790.2
63	local copy	remote version	590 Animals				✓	599.5094678
			310 Collections of General Statistics	✓				

64	local copy	remote version	070.1 Documentary media, educational media, news media	✓			✓	~~004.6~~ ~~070.21~~ 070.19
			004.6 Interfacing and communications (Computer science)	✓	✓			
65	local copy	remote version	750 Painting and Paintings	✓				332.6
			660 Chemical Engineering	✓				
66	local copy	remote version	380 Commerce, Communications, Transportation	✓				264.033
			320 Political Science	✓				
67	local copy	remote version	004.7 Peripherals (Computer science)		✓			704.9 uncertain if correct
			410 Linguistics	✓				
68	local copy	remote version	650 Management and Auxiliary Services	✓				790.079
			640 Home Economics and Family Living	✓				
69	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004.6
			005.1 Programming (Computer programming)			✓		
70	local copy	remote version	370 Education				✓	370.285
			520 Astronomy and Allied Sciences	✓				
71	local copy	remote version	370 Education				✓	370.285
72	local copy	remote version	320 Political Science		✓			306.766
			305 Social Groups			✓		
73	local copy	remote version	660 Chemical Engineering	✓				004.6
			380 Commerce, Communications, Transportation	✓				
74	local copy	remote version	660 Chemical Engineering		✓			354.44
			650 Management and Auxiliary Services	✓				
75	local copy	remote version	027 General libraries, archives, information centres			✓		025.5
			025.5 Services to users (Operations of libraries, archives, information centres				✓	
76	local copy	remote version	790 Recreational and Performing Arts		✓			378.166
			780 Music		✓			
77	local copy	remote version	380 Commerce, Communications, Transportation		✓			338.7
78	local copy	remote version	650 Management and Auxiliary Services					not enough information
					✓			
79	local copy	remote version	340 Law	✓				333.7
			350 Public Administration and Military Science	✓				

80	local copy	remote version	004.0151 Mathematical principles (Computer Systems)				✓	510
81	local copy	remote version	005.7 Data in computer systems				✓	005.7
			004.1 General works on specific types of computers			✓		
82	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004.6
			003.7 Kinds of Systems (Computer Systems)			✓		
83	local copy	remote version	780 Music			✓		621.3893
			250 Christian Orders and Local Church	✓				
84	local copy	remote version	307 Communities				✓	307
			305 Social Groups			✓		
85	local copy	remote version	370 Education		✓			331.124
			340 Law	✓				
86	local copy	remote version	150 Psychology		✓			006.30711
			005.1 Programming (Computer programming)			✓		
87	local copy	remote version	027 General libraries, archives, information centres		✓			335.4
			004.6 Interfacing and communications (Computer science)		✓			
88	local copy	remote version	004.1 General works on specific types of computers			✓		005.4
			005.4 Systems programming and programs (Computer programs)				✓	
89	local copy	remote version	790 Recreational and Performing Arts				✓	793.92
			710 Civic and landscape Art	✓				
90	local copy	remote version	650 Management and Auxiliary Services	✓				378.1
			330 Economics	✓				
91	local copy	remote version	750 Painting and Paintings	✓				332.6
			660 Chemical Engineering	✓				
92	local copy	remote version	070.1 Documentary media, educational media, news media	✓				551.654167
			790 Recreational and Performing Arts	✓				
93	local copy	remote version	690 Buildings			✓		320.6
			320 Political Science				✓	
94	local copy	remote version	380 Commerce, Communications, Transportation				✓	388.4132
95	local copy	remote version	620 Engineering and Allied Operations				✓	629.10711
			003.7 Kinds of Systems (Computer Systems)	✓				

96	local copy	remote version	005.7 Data in computer systems		✓			790
			790 Recreational and Performing Arts				✓	
97	local copy	remote version	027 General libraries, archives, information centres			✓		025.5
			025.5 Services to users (Operations of libraries, archives, information centres				✓	
98	local copy	remote version	370 Education		✓			025.5
			350 Public Administration and Military Science	✓				
99	local copy	remote version	370 Education				✓	371.26
100	local copy	remote version	620 Engineering and Allied Operations	✓				323.64
			310 Collections of General Statistics	✓				
101	local copy	remote version	760 Graphic Arts Printmaking Prints				✓	760
102	local copy	remote version	660 Chemical Engineering	✓				338.911
			640 Home Economics and Family Living	✓				
103	local copy	remote version	670 Manufacturing	✓				025.5202854
			640 Home Economics and Family Living	✓				
104	local copy	remote version	610 Medical Sciences, Medicine				✓	610
			025.5 Services to users (Operations of libraries, archives, information centres	✓				
105	local copy	remote version	005.1 Programming (Computer programming)		✓			745.40285
			004.6 Interfacing and communications (Computer science)		✓			
106	local copy	remote version	370 Education			✓		004.071142293
			690 Buildings	✓				
107	local copy	remote version	690 Buildings				✓	690
108	local copy	remote version	640 Home Economics and Family Living	✓				914.235
			790 Recreational and Performing Arts	✓				
109	local copy	remote version	640 Home Economics and Family Living			✓		388.044
110	local copy	remote version	640 Home Economics and Family Living	✓				004.6
			004.6 Interfacing and communications (Computer science)				✓	
111	local copy	remote version	380 Commerce, Communications, Transportation				✓	388.413214
			640 Home Economics and Family Living	✓				

112	local copy	remote version	070.1 Documentary media, educational media, news media	✓				387.542
			380 Commerce, Communications, Transportation				✓	
113	local copy	remote version	006.7 Multimedia (Special computer methods	✓				381.142
			006.6 Computer graphics	✓				
114	local copy	remote version	380 Commerce, Communications, Transportation	✓				646.770285
			320 Political Science	✓				
115	local copy	remote version	620 Engineering and Allied Operations	✓				796.79
			790 Recreational and Performing Arts				✓	
116	local copy	remote version	530 Physics				✓	530
			520 Astronomy and Allied Sciences	✓				
117	local copy	remote version	790 Recreational and Performing Arts			✓		371.242
			780 Music			✓		
118	local copy	remote version	302 Social Interaction	✓				200.71
			120 Epistemology, Causation, Humankind	✓				
119	local copy	remote version	620 Engineering and Allied Operations	✓				910.202
			310 Collections of General Statistics	✓				
120	local copy	remote version	320 Political Science				✓	320.71
			490 Other Languages	✓				
121	local copy	remote version	530 Physics			✓		507.1
			005.7 Data in computer systems	✓				
122	local copy	remote version	350 Public Administration and Military Science		✓			~~600~~ ~~700~~ 507.1
			005.8 Data security		✓			
123	local copy	remote version	070.1 Documentary media, educational media, news media		✓			384.50285
			780 Music	✓				
124	local copy	remote version	780 Music				✓	780.79
			070.1 Documentary media, educational media, news media	✓				
125	local copy	remote version	250 Christian Orders and Local Church		✓			726.5
			230 Christianity Christian Theology		✓			
126	local copy	remote version	320 Political Science				✓	324.24104
			070.1 Documentary media, educational media, news media	✓				
127	local copy	remote version	380 Commerce, Communications, Transportation				✓	388.342
			590 Animals	✓				

128	local copy	remote version	005.7 Data in computer systems		✓			384.6
			380 Commerce, Communications, Transportation				✓	
129	local copy	remote version	790 Recreational and Performing Arts				✓	796.352068
			780 Music	✓				
130	local copy	remote version	790 Recreational and Performing Arts				✓	796
			380 Commerce, Communications, Transportation	✓				
131	local copy	remote version	590 Animals			✓		808.838762
			004.6 Interfacing and communications (Computer science)	✓				
132	local copy	remote version	370 Education				✓	378.101
133	local copy	remote version	590 Animals	✓				914.129
			780 Music	✓				
134	local copy	remote version	350 Public Administration and Military Science		✓			387.73
			006.3 Artificial intelligence	✓				
135	local copy	remote version	790 Recreational and Performing Arts		✓			381.142
			004.1 General works on specific types of computers	✓				
136	local copy	remote version	004.6 Interfacing and communications (Computer science)	✓				371.33
			370 Education				✓	
137	local copy	remote version	006.6 Computer graphics				✓	006.6
			750 Painting and Paintings			✓		
138	local copy	remote version	350 Public Administration and Military Science	✓				372.216
139	local copy	remote version	004.6 Interfacing and communications (Computer science)	✓				942.982
			508 Natural history		✓			
140	local copy	remote version	570 Life Sciences, Biology			✓		501
			530 Physics			✓		
141	local copy	remote version	610 Medical Sciences, Medicine				✓	616.12
			370 Education	✓				
142	local copy	remote version	380 Commerce, Communications, Transportation	✓				004:092
			310 Collections of General Statistics	✓				
143	local copy	remote version	790 Recreational and Performing Arts				✓	796.34244258
			340 Law	✓				

144	local copy	remote version	004.7 Peripherals (Computer science)	✓	✓			914. 6804
			190 Modern Western Philosophy	✓				
145	local copy	remote version	660 Chemical Engineering	✓				363.728
			650 Management and Auxiliary Services	✓				
146	local copy	remote version	790 Recreational and Performing Arts					not enough information
147	local copy	remote version	640 Home Economics and Family Living	✓				914.235
			790 Recreational and Performing Arts	✓				
148	local copy	remote version	380 Commerce, Communications, Transportation	✓				919.49504
149	local copy	remote version	780 Music				✓	781.642
			070.1 Documentary media, educational media, news media	✓				
150	local copy	remote version	380 Commerce, Communications, Transportation				✓	388.342
151	local copy	remote version	790 Recreational and Performing Arts				✓	796
			070.1 Documentary media, educational media, news media			✓		
152	local copy	remote version	670 Manufacturing	✓				004.092
			380 Commerce, Communications, Transportation	✓				
153	local copy	remote version	510 Mathematics				✓	516
154	local copy	remote version	004.6 Interfacing and communications (Computer science)			✓		005.3
			005.1 Programming (Computer programming)				✓	
155	local copy	remote version	370 Education				✓	370.113
156	local copy	remote version	370 Education			✓		004.0711
			650 Management and Auxiliary Services	✓				
157	local copy	remote version	780 Music				✓	784.2
			790 Recreational and Performing Arts		✓			
158	local copy	remote version	380 Commerce, Communications, Transportation		✓			333.917
159	local copy	remote version	350 Public Administration and Military Science		✓			361.77
			307 Communities		✓			

160	local copy	remote version	025.5 Services to users (Operations of libraries, archives, information centres				✓	025.5
			005.7 Data in computer systems		✓			
161	local copy	remote version	350 Public Administration and Military Science	✓				070.1
			740 Drawing and Decorative Arts	✓				
162	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004.6
			380 Commerce, Communications, Transportation	✓				
163	local copy	remote version	004.0151 Mathematical principles (Computer Systems)	✓				331.702
164	local copy	remote version	027 General libraries, archives, information centres				✓	027
			690 Buildings	✓				
165	local copy	remote version	380 Commerce, Communications, Transportation		✓			359.4
			307 Communities	✓				
166	local copy	remote version	640 Home Economics and Family Living				✓	641.5
			025.5 Services to users (Operations of libraries, archives, information centres	✓				
167	local copy	remote version	070.1 Documentary media, educational media, news media			✓		914.68
			070.4 Journalism			✓		
168	local copy	remote version	650 Management and Auxiliary Services	✓				709.4227
169	local copy	remote version	650 Management and Auxiliary Services	✓				331.124
			750 Painting and Paintings	✓				
170	local copy	remote version	370 Education	✓				363.20715
			650 Management and Auxiliary Services		✓			
171	local copy	remote version	370 Education				✓	378.199
			650 Management and Auxiliary Services	✓				
172	local copy	remote version	640 Home Economics and Family Living	✓				387.2
			790 Recreational and Performing Arts			✓		
173	local copy	remote version	690 Buildings	✓				378.19
			380 Commerce, Communications, Transportation	✓				
174	local copy	remote version	690 Buildings		✓			645.3
			680 Manufacture for Specific uses			✓		
175	local copy	remote version	370 Education				✓	371.334
			004.6 Interfacing and communications (Computer science)			✓		

176	local copy	remote version	004.6 Interfacing and communications (Computer science)		✓			028.7
			005.7 Data in computer systems		✓			
177	local copy	remote version	005.7 Data in computer systems		✓			028.7
178	local copy	remote version	005.7 Data in computer systems				✓	005.741
			025.5 Services to users (Operations of libraries, archives, information centres			✓		
179	local copy	remote version	340 Law				✓	346.048071
			027 General libraries, archives, information centres	✓				
180	local copy	remote version	370 Education			✓		331.124
			650 Management and Auxiliary Services	✓				
181	local copy	remote version	790 Recreational and Performing Arts				✓	796.33
			680 Manufacture for Specific uses	✓				
182	local copy	remote version	720 Architecture		✓			709.468
			708 Galleries, Museums, Private Collections			✓		
183	local copy	remote version	780 Music	✓		✓		791.447
			070.1 Documentary media, educational media, news media		✓			
184	local copy	remote version	780 Music				✓	781.63
			070.1 Documentary media, educational media, news media		✓			
185	local copy	remote version	370 Education				✓	378.19
			330 Economics		✓			
186	local copy	remote version	370 Education		✓			646.7240715
			350 Public Administration and Military Science	✓				
187	local copy	remote version	027 General libraries, archives, information centres				✓	027
			025.8 Maintenance and preservation (Operations of libraries, archives, information centres		✓			
188	local copy	remote version	305 Social Groups	✓				616.852 7061
			610 Medical Sciences, Medicine				✓	
189	local copy	remote version	190 Modern Western Philosophy			✓		291.436
			110 Metaphysics			✓		
190	local copy	remote version	370 Education				✓	372.9
191	local copy	remote version	380 Commerce, Communications, Transportation	✓				658.404 unsure because of infomation

192	local copy	remote version	670 Manufacturing	✓				616.853
			610 Medical Sciences, Medicine				✓	
193	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004.6
			004.3 Processing modes (Computer science)			✓		
194	local copy	remote version	690 Buildings			✓		610.71141443
			022 Administration of the physical plant (Library and information sciences)	✓				
195	local copy	remote version	307 Communities		✓			361.77
			790 Recreational and Performing Arts			✓		
196	local copy	remote version	370 Education	✓				025.5
			021 Relationships of libraries, archives, information centres				✓	
197	local copy	remote version	790 Recreational and Performing Arts			✓		914.404
			750 Painting and Paintings	✓				
198	local copy	remote version	340 Law	✓				378.19
			380 Commerce, Communications, Transportation	✓				
199	local copy	remote version	790 Recreational and Performing Arts				✓	793.932
200	local copy	remote version	650 Management and Auxiliary Services	✓				363.2094227
			380 Commerce, Communications, Transportation	✓				

Classifier Results *Librarian 2*

No.	Local	Remote	Classmarks	Rating 1	2	3	4	Manual classification
1	local copy	remote version	370 Education				✓	378
2	local copy	remote version	640 Home Economics and Family Living		✓			696·1
3	local copy	remote version	004.6 Interfacing and communications (Computer science)			✓		005·72
			005.7 Data in computer systems				✓	
4	local copy	remote version	640 Home Economics and Family Living				✓	647
			790 Recreational and Performing Arts			✓		
5	local copy	remote version	320 Political Science				✓	321
			190 Modern Western Philosophy			✓		
6	local copy	remote version	070.1 Documentary media, educational media, news media				✓	070·1
			340 Law		✓			
7	local copy	remote version	690 Buildings				✓	690
			640 Home Economics and Family Living				✓	
8	local copy	remote version	640 Home Economics and Family Living		✓			380·1
			380 Commerce, Communications, Transportation				✓	
9	local copy	remote version	380 Commerce, Communications, Transportation	✓				378·11
10	local copy	remote version	370 Education				✓	372
			304 Social Behaviour		✓			
11	local copy	remote version	510 Mathematics				✓	507·11
			150 Psychology	✓				
12	local copy	remote version	330 Economics		✓			650·14
			650 Management and Auxiliary Services				✓	
13	local copy	remote version	708 Galleries, Museums, Private Collections				✓	708
			069 Museology (Museum science)			✓		
14	local copy	remote version	610 Medical Sciences, Medicine				✓	610·711
			370 Education				✓	
15	local copy	remote version	520 Astronomy and Allied Sciences	✓				600
16	local copy	remote version	307 Communities				✓	307
			305 Social Groups			✓		

17	local copy	remote version	003.5 Theory of communication and control (Computer Systems)				✓	003·5
18	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004·6
			680 Manufacture for Specific uses				✓		
19	local copy	remote version	370 Education				✓	370
			021 Relationships of libraries, archives, information centres			✓			
20	local copy	remote version	790 Recreational and Performing Arts				✓	005·3
			005.3 Programs (Computer programs)				✓		
21	local copy	remote version	370 Education				✓	374
			021 Relationships of libraries, archives, information centres			✓			
22	local copy	remote version	380 Commerce, Communications, Transportation	✓				365
23	local copy	remote version	650 Management and Auxiliary Services			✓		020
			027 General libraries, archives, information centres			✓			
24	local copy	remote version	590 Animals		✓			133
			690 Buildings			✓			
25	local copy	remote version	340 Law	✓				643
26	local copy	remote version	670 Manufacturing	✓				005·7
			005.7 Data in computer systems				✓		
27	local copy	remote version	610 Medical Sciences, Medicine				✓	610
			370 Education			✓			
28	local copy	remote version	005.7 Data in computer systems				✓	005·7
			005.1 Programming (Computer programming)			✓			
29	local copy	remote version	070.1 Documentary media, educational media, news media				✓	070·1
			070.4 Journalism				✓		
30	local copy	remote version	027 General libraries, archives, information centres				✓	027
			690 Buildings		✓				
31	local copy	remote version	340 Law				✓	346
32	local copy	remote version	370 Education				✓	371·2
			650 Management and Auxiliary Services			✓			
33	local copy	remote version	070.1 Documentary media, educational media, news media		✓			380·1
			350 Public Administration and Military Science	✓					

34	local copy	remote version	820 English and Old English Literatures				✓	380.1
			790 Recreational and Performing Arts			✓		
35	local copy	remote version	027 General libraries, archives, information centres			✓		378
			022 Administration of the physical plant (Library and information sciences)		✓			
36	local copy	remote version	790 Recreational and Performing Arts		✓			355
			005.8 Data security			✓		
37	local copy	remote version	670 Manufacturing	✓				004
			380 Commerce, Communications, Transportation	✓				
38	local copy	remote version	790 Recreational and Performing Arts				✓	338.4791
			004.6 Interfacing and communications (Computer science)	✓				
39	local copy	remote version	780 Music				✓	784
			005.1 Programming (Computer programming)	✓				
40	local copy	remote version	027 General libraries, archives, information centres				✓	335
			420 English	✓				
41	local copy	remote version	790 Recreational and Performing Arts				✓	790.1
			780 Music	✓				
42	local copy	remote version	004.1 General works on specific types of computers				✓	004.1
			005.4 Systems programming and programs (Computer programs)			✓		
43	local copy	remote version	790 Recreational and Performing Arts				✓	780
			780 Music				✓	
44	local copy	remote version	790 Recreational and Performing Arts				✓	796
45	local copy	remote version	307 Communities				✓	307
			305 Social Groups				✓	
46	local copy	remote version	370 Education				✓	570.7
			350 Public Administration and Military Science	✓				
47	local copy	remote version	510 Mathematics				✓	378
			370 Education				✓	
48	local copy	remote version	025.5 Services to users (Operations of libraries, archives, information centres				✓	025.5
			027 General libraries, archives, information centres			✓		
49	local copy	remote version	070.1 Documentary media, educational media, news media	✓				647
50	local	remote	370 Education				✓	372
			027 General libraries, archives, information			✓		

	copy	version	centres					
51	local copy	remote version	027 General libraries, archives, information centres				✓	027
			610 Medical Sciences, Medicine	✓				
52	local copy	remote version	340 Law			✓		327
			320 Political Science				✓	
53	local copy	remote version	025.5 Services to users (Operations of libraries, archives, information centres				✓	025·5
			380 Commerce, Communications, Transportation	✓				
54	local copy	remote version	004.1 General works on specific types of computers				✓	025·5
			027 General libraries, archives, information centres				✓	
55	local copy	remote version	340 Law	✓				647
			630 Agriculture and Related Technologies	✓				
56	local copy	remote version	370 Education				✓	004·6
			004.6 Interfacing and communications (Computer science)				✓	
57	local copy	remote version	750 Painting and Paintings	✓				647
			740 Drawing and Decorative Arts	✓				
58	local copy	remote version	370 Education				✓	378
			530 Physics				✓	
59	local copy	remote version	340 Law				✓	378
			370 Education				✓	
60	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004·6
			005.3 Programs (Computer programs)				✓	
61	local copy	remote version	590 Animals	✓				005·741
62	local copy	remote version	820 English and Old English Literatures	✓				011·7
63	local copy	remote version	590 Animals				✓	590
			310 Collections of General Statistics		✓			
64	local copy	remote version	070.1 Documentary media, educational media, news media				✓	070·1
			004.6 Interfacing and communications (Computer science)				✓	
65	local copy	remote version	750 Painting and Paintings	✓				332·6
			660 Chemical Engineering	✓				
66	local copy	remote version	380 Commerce, Communications, Transportation				✓	380·1
			320 Political Science	✓				
67	local copy	remote version	004.7 Peripherals (Computer science)				✓	004·7
			410 Linguistics	✓				

68	local copy	remote version	650 Management and Auxiliary Services	✓				011·7
			640 Home Economics and Family Living	✓				
69	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004·6
			005.1 Programming (Computer programming)			✓		
70	local copy	remote version	370 Education				✓	370
			520 Astronomy and Allied Sciences	✓				
71	local copy	remote version	370 Education				✓	378
72	local copy	remote version	320 Political Science					305
			305 Social Groups				✓	
73	local copy	remote version	660 Chemical Engineering	✓				004·6
			380 Commerce, Communications, Transportation				✓	
74	local copy	remote version	660 Chemical Engineering				✓	660
			650 Management and Auxiliary Services			✓		
75	local copy	remote version	027 General libraries, archives, information centres				✓	025·5
			025.5 Services to users (Operations of libraries, archives, information centres				✓	
76	local copy	remote version	790 Recreational and Performing Arts	✓				378
			780 Music	✓				
77	local copy	remote version	380 Commerce, Communications, Transportation				✓	380·1
78	local copy	remote version	650 Management and Auxiliary Services	✓				796
79	local copy	remote version	340 Law				✓	344
			350 Public Administration and Military Science				✓	
80	local copy	remote version	004.0151 Mathematical principles (Computer Systems)			✓		510
81	local copy	remote version	005.7 Data in computer systems				✓	005·7
			004.1 General works on specific types of computers				✓	
82	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004·6
			003.7 Kinds of Systems (Computer Systems)	✓				
83	local copy	remote version	780 Music				✓	780
			250 Christian Orders and Local Church	✓				
84	local copy	remote version	307 Communities				✓	307
			305 Social Groups			✓		

85	local copy	remote version	370 Education				✓	331
			340 Law	✓				
86	local copy	remote version	150 Psychology				✓	378
			005.1 Programming (Computer programming)				✓	
87	local copy	remote version	027 General libraries, archives, information centres				✓	335
			004.6 Interfacing and communications (Computer science)			✓		
88	local copy	remote version	004.1 General works on specific types of computers				✓	004.1
			005.4 Systems programming and programs (Computer programs)				✓	
89	local copy	remote version	790 Recreational and Performing Arts				✓	796
			710 Civic and landscape Art	✓				
90	local copy	remote version	650 Management and Auxiliary Services				✓	658.3
			330 Economics	✓				
91	local copy	remote version	750 Painting and Paintings	✓				332.6
			660 Chemical Engineering	✓				
92	local copy	remote version	070.1 Documentary media, educational media, news media				✓	070.1
			790 Recreational and Performing Arts	✓				
93	local copy	remote version	690 Buildings		✓			363.7
			320 Political Science			✓		
94	local copy	remote version	380 Commerce, Communications, Transportation				✓	380.1
95	local copy	remote version	620 Engineering and Allied Operations				✓	378
			003.7 Kinds of Systems (Computer Systems)	✓				
96	local copy	remote version	005.7 Data in computer systems				✓	005.7
			790 Recreational and Performing Arts			✓		
97	local copy	remote version	027 General libraries, archives, information centres				✓	025.5
			025.5 Services to users (Operations of libraries, archives, information centres				✓	
98	local copy	remote version	370 Education				✓	025.5
			350 Public Administration and Military Science	✓				
99	local copy	remote version	370 Education				✓	370
100	local copy	remote version	620 Engineering and Allied Operations	✓				323
			310 Collections of General Statistics	✓				
101	local copy	remote version	760 Graphic Arts Printmaking Prints				✓	760

102	local copy	remote version	660 Chemical Engineering	✓				361.7
			640 Home Economics and Family Living			✓		
103	local copy	remote version	670 Manufacturing		✓			387
			640 Home Economics and Family Living	✓				
104	local copy	remote version	610 Medical Sciences, Medicine				✓	610
			025.5 Services to users (Operations of libraries, archives, information centres			✓		
105	local copy	remote version	005.1 Programming (Computer programming)			✓		004.6
			004.6 Interfacing and communications (Computer science)				✓	
106	local copy	remote version	370 Education				✓	378
			690 Buildings	✓				
107	local copy	remote version	690 Buildings		✓			647
108	local copy	remote version	640 Home Economics and Family Living				✓	647
			790 Recreational and Performing Arts				✓	
109	local copy	remote version	640 Home Economics and Family Living					388
110	local copy	remote version	640 Home Economics and Family Living				✓	004.6
			004.6 Interfacing and communications (Computer science)				✓	
111	local copy	remote version	380 Commerce, Communications, Transportation			✓		338.4791
			640 Home Economics and Family Living			✓		
112	local copy	remote version	070.1 Documentary media, educational media, news media				✓	070.1
			380 Commerce, Communications, Transportation				✓	
113	local copy	remote version	006.7 Multimedia (Special computer methods		✓			380.1
			006.6 Computer graphics		✓			
114	local copy	remote version	380 Commerce, Communications, Transportation				✓	306
			320 Political Science	✓				
115	local copy	remote version	620 Engineering and Allied Operations	✓				790
			790 Recreational and Performing Arts				✓	
116	local copy	remote version	530 Physics				✓	530
			520 Astronomy and Allied Sciences				✓	
117	local copy	remote version	790 Recreational and Performing Arts		✓			370
			780 Music	✓				
118	local copy	remote version	302 Social Interaction	✓				374
			120 Epistemology, Causation, Humankind			✓		
119	local copy	remote version	620 Engineering and Allied Operations	✓				910
			310 Collections of General Statistics			✓		

120	local copy	remote version	320 Political Science				✓	374
			490 Other Languages	✓				
121	local copy	remote version	530 Physics				✓	378
			005.7 Data in computer systems		✓			
122	local copy	remote version	350 Public Administration and Military Science				✓	507
			005.8 Data security	✓				
123	local copy	remote version	070.1 Documentary media, educational media, news media				✓	070·1
			780 Music		✓			
124	local copy	remote version	780 Music				✓	306
			070.1 Documentary media, educational media, news media				✓	
125	local copy	remote version	250 Christian Orders and Local Church				✓	250
			230 Christianity Christian Theology				✓	
126	local copy	remote version	320 Political Science				✓	320
			070.1 Documentary media, educational media, news media				✓	
127	local copy	remote version	380 Commerce, Communications, Transportation				✓	380
			590 Animals	✓				
128	local copy	remote version	005.7 Data in computer systems				✓	384
			380 Commerce, Communications, Transportation				✓	
129	local copy	remote version	790 Recreational and Performing Arts				✓	790
			780 Music		✓			
130	local copy	remote version	790 Recreational and Performing Arts				✓	790
			380 Commerce, Communications, Transportation				✓	
131	local copy	remote version	590 Animals	✓				070·1
			004.6 Interfacing and communications (Computer science)			✓		
132	local copy	remote version	370 Education				✓	378
133	local copy	remote version	590 Animals	✓				338·4791
			780 Music	✓				
134	local copy	remote version	350 Public Administration and Military Science				✓	387
			006.3 Artificial intelligence	✓				
135	local copy	remote version	790 Recreational and Performing Arts				✓	790
			004.1 General works on specific types of computers				✓	
136	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	370
			370 Education				✓	

137	local copy	remote version	006.6 Computer graphics				✓	006.6
			750 Painting and Paintings				✓	
138	local copy	remote version	350 Public Administration and Military Science	✓				370
139	local copy	remote version	004.6 Interfacing and communications (Computer science)	✓				508
			508 Natural history				✓	
140	local copy	remote version	570 Life Sciences, Biology				✓	570
			530 Physics				✓	
141	local copy	remote version	610 Medical Sciences, Medicine				✓	610
			370 Education		✓			
142	local copy	remote version	380 Commerce, Communications, Transportation					001.4 OR 378
			310 Collections of General Statistics	✓				
143	local copy	remote version	790 Recreational and Performing Arts				✓	796
			340 Law	✓				
144	local copy	remote version	004.7 Peripherals (Computer science)		✓			946
			190 Modern Western Philosophy	✓				
145	local copy	remote version	660 Chemical Engineering	✓				620
			650 Management and Auxiliary Services	✓				
146	local copy	remote version	790 Recreational and Performing Arts				✓	796
147	local copy	remote version	640 Home Economics and Family Living		✓			647
			790 Recreational and Performing Arts				✓	
148	local copy	remote version	380 Commerce, Communications, Transportation				✓	338.4791
149	local copy	remote version	780 Music				✓	780
			070.1 Documentary media, educational media, news media	✓				
150	local copy	remote version	380 Commerce, Communications, Transportation				✓	380.1
151	local copy	remote version	790 Recreational and Performing Arts				✓	790
			070.1 Documentary media, educational media, news media				✓	
152	local copy	remote version	670 Manufacturing	✓				378
			380 Commerce, Communications, Transportation	✓				
153	local copy	remote version	510 Mathematics				✓	510

154	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓		005.1
			005.1 Programming (Computer programming)					✓	
155	local copy	remote version	370 Education					✓	378
156	local copy	remote version	370 Education					✓	378
			650 Management and Auxiliary Services				✓		
157	local copy	remote version	780 Music					✓	780
			790 Recreational and Performing Arts					✓	
158	local copy	remote version	380 Commerce, Communications, Transportation		✓				551
159	local copy	remote version	350 Public Administration and Military Science				✓		360
			307 Communities					✓	
160	local copy	remote version	025.5 Services to users (Operations of libraries, archives, information centres					✓	025.5
			005.7 Data in computer systems					✓	
161	local copy	remote version	350 Public Administration and Military Science					✓	355
			740 Drawing and Decorative Arts		✓				
162	local copy	remote version	004.6 Interfacing and communications (Computer science)					✓	004.6
			380 Commerce, Communications, Transportation					✓	
163	local copy	remote version	004.0151 Mathematical principles (Computer Systems)				✓		331
164	local copy	remote version	027 General libraries, archives, information centres					✓	027
			690 Buildings		✓				
165	local copy	remote version	380 Commerce, Communications, Transportation			✓			942
			307 Communities					✓	
166	local copy	remote version	640 Home Economics and Family Living					✓	640
			025.5 Services to users (Operations of libraries, archives, information centres			✓			
167	local copy	remote version	070.1 Documentary media, educational media, news media					✓	070.1
			070.4 Journalism					✓	
168	local copy	remote version	650 Management and Auxiliary Services		✓				910
169	local copy	remote version	650 Management and Auxiliary Services				✓		331
			750 Painting and Paintings		✓				

170	local copy	remote version	370 Education				✓	378
			650 Management and Auxiliary Services		✓			
171	local copy	remote version	370 Education				✓	378
			650 Management and Auxiliary Services	✓				
172	local copy	remote version	640 Home Economics and Family Living			✓		380·1
			790 Recreational and Performing Arts			✓		
173	local copy	remote version	690 Buildings	✓				378
			380 Commerce, Communications, Transportation	✓				
174	local copy	remote version	690 Buildings	✓				680
			680 Manufacture for Specific uses				✓	
175	local copy	remote version	370 Education				✓	004·6
			004.6 Interfacing and communications (Computer science)				✓	
176	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004·6
			005.7 Data in computer systems			✓		
177	local copy	remote version	005.7 Data in computer systems				✓	005·7
178	local copy	remote version	005.7 Data in computer systems				✓	005·7
			025.5 Services to users (Operations of libraries, archives, information centres		✓			
179	local copy	remote version	340 Law				✓	378
			027 General libraries, archives, information centres		✓			
180	local copy	remote version	370 Education				✓	370
			650 Management and Auxiliary Services		✓			
181	local copy	remote version	790 Recreational and Performing Arts				✓	796
			680 Manufacture for Specific uses	✓				
182	local copy	remote version	720 Architecture			✓		700
			708 Galleries, Museums, Private Collections			✓		
183	local copy	remote version	780 Music		✓			790
			070.1 Documentary media, educational media, news media		✓			
184	local copy	remote version	780 Music				✓	070·1
			070.1 Documentary media, educational media, news media			✓		
185	local copy	remote version	370 Education				✓	378
			330 Economics			✓		
186	local copy	remote version	370 Education				✓	374
			350 Public Administration and Military Science	✓				

187	local copy	remote version	027 General libraries, archives, information centres				✓	025.8
			025.8 Maintenance and preservation (Operations of libraries, archives, information centres				✓	
188	local copy	remote version	305 Social Groups		✓			610
			610 Medical Sciences, Medicine				✓	
189	local copy	remote version	190 Modern Western Philosophy				✓	190
			110 Metaphysics		✓			
190	local copy	remote version	370 Education				✓	372
191	local copy	remote version	380 Commerce, Communications, Transportation			✓		658.4
192	local copy	remote version	670 Manufacturing	✓				610
			610 Medical Sciences, Medicine				✓	
193	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004.6
			004.3 Processing modes (Computer science)		✓			
194	local copy	remote version	690 Buildings				✓	378
			022 Administration of the physical plant (Library and information sciences)	✓				
195	local copy	remote version	307 Communities				✓	360
			790 Recreational and Performing Arts		✓			
196	local copy	remote version	370 Education				✓	370
			021 Relationships of libraries, archives, information centres				✓	
197	local copy	remote version	790 Recreational and Performing Arts				✓	647
			750 Painting and Paintings	✓				
198	local copy	remote version	340 Law	✓				378
			380 Commerce, Communications, Transportation	✓				
199	local copy	remote version	790 Recreational and Performing Arts				✓	790
200	local copy	remote version	650 Management and Auxiliary Services		✓			910
			380 Commerce, Communications, Transportation	✓				

Classifier Results *Librarian 3*

No.	Local	Remote	Classmarks	Rating 1	2	3	4	Manual classification
1	local copy	remote version	370 Education				✓	378
2	local copy	remote version	640 Home Economics and Family Living	✓				696.1
3	local copy	remote version	004.6 Interfacing and communications (Computer science)			✓		005.72
			005.7 Data in computer systems				✓	
4	local copy	remote version	640 Home Economics and Family Living				✓	647.95
			790 Recreational and Performing Arts			✓		
5	local copy	remote version	320 Political Science				✓	324
			190 Modern Western Philosophy			✓		
6	local copy	remote version	070.1 Documentary media, educational media, news media	✓				004.6
			340 Law	✓				
7	local copy	remote version	690 Buildings	✓				642.12
			640 Home Economics and Family Living				✓	
8	local copy	remote version	640 Home Economics and Family Living	✓				338.71
			380 Commerce, Communications, Transportation			✓		
9	local copy	remote version	380 Commerce, Communications, Transportation	✓				378
10	local copy	remote version	370 Education				✓	370.115
			304 Social Behaviour			✓		
11	local copy	remote version	510 Mathematics			✓		510
			150 Psychology	✓				
12	local copy	remote version	330 Economics				✓	657.8
			650 Management and Auxiliary Services				✓	
13	local copy	remote version	708 Galleries, Museums, Private Collections				✓	708
			069 Museology (Museum science)		✓			
14	local copy	remote version	610 Medical Sciences, Medicine				✓	610.7
			370 Education			✓		
15	local copy	remote version	520 Astronomy and Allied Sciences	✓				338.926
16	local copy	remote version	307 Communities				✓	307.34
			305 Social Groups			✓		

17	local copy	remote version	003.5 Theory of communication and control (Computer Systems)				✓	002.5
18	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004.6
			680 Manufacture for Specific uses	✓				
19	local copy	remote version	370 Education			✓		021.65
			021 Relationships of libraries, archives, information centres				✓	
20	local copy	remote version	790 Recreational and Performing Arts				✓	790.1
			005.3 Programs (Computer programs)				✓	
21	local copy	remote version	370 Education				✓	371.9
			021 Relationships of libraries, archives, information centres			✓		
22	local copy	remote version	380 Commerce, Communications, Transportation	✓				321.851
23	local copy	remote version	650 Management and Auxiliary Services			✓		027.007
			027 General libraries, archives, information centres				✓	
24	local copy	remote version	590 Animals	✓				123.3327
			690 Buildings	✓				
25	local copy	remote version	340 Law	✓				328.4791
26	local copy	remote version	670 Manufacturing	✓				700.9
			005.7 Data in computer systems	✓				
27	local copy	remote version	610 Medical Sciences, Medicine				✓	610.7
			370 Education		✓			
28	local copy	remote version	005.7 Data in computer systems				✓	660.6 (difficult information)
			005.1 Programming (Computer programming)				✓	
29	local copy	remote version	070.1 Documentary media, educational media, news media					005.7
			070.4 Journalism					
30	local copy	remote version	027 General libraries, archives, information centres				✓	027.2
			690 Buildings	✓				
31	local copy	remote version	340 Law				✓	346.077
32	local copy	remote version	370 Education				✓	371.2
			650 Management and Auxiliary Services		✓			

33	local copy	remote version	070.1 Documentary media, educational media, news media	✓				791.4375
			350 Public Administration and Military Science	✓				
34	local copy	remote version	820 English and Old English Literatures			✓		381.45002
			790 Recreational and Performing Arts	✓				
35	local copy	remote version	027 General libraries, archives, information centres	✓				378
			022 Administration of the physical plant (Library and information sciences)	✓				
36	local copy	remote version	790 Recreational and Performing Arts	✓				352.379
			005.8 Data security	✓				
37	local copy	remote version	670 Manufacturing	✓				650.14
			380 Commerce, Communications, Transportation	✓				
38	local copy	remote version	790 Recreational and Performing Arts	✓				338.4791
			004.6 Interfacing and communications (Computer science)	✓				
39	local copy	remote version	780 Music				✓	784.2
			005.1 Programming (Computer programming)	✓				
40	local copy	remote version	027 General libraries, archives, information centres			✓		335.4
			420 English	✓				
41	local copy	remote version	790 Recreational and Performing Arts				✓	792
			780 Music		✓			
42	local copy	remote version	004.1 General works on specific types of computers		✓			331 128
			005.4 Systems programming and programs (Computer programs)		✓			
43	local copy	remote version	790 Recreational and Performing Arts			✓		780.905
			780 Music				✓	
44	local copy	remote version	790 Recreational and Performing Arts				✓	796.81C2
45	local copy	remote version	307 Communities				✓	307
			305 Social Groups		✓			
46	local copy	remote version	370 Education				✓	370.7
			350 Public Administration and Military Science		✓			
47	local copy	remote version	510 Mathematics				✓	510
			370 Education			✓		

48	local copy	remote version	025.5 Services to users (Operations of libraries, archives, information centres	✓				384
			027 General libraries, archives, information centres	✓				
49	local copy	remote version	070.1 Documentary media, educational media, news media	✓				338.4791
50	local copy	remote version	370 Education				✓	370.7
			027 General libraries, archives, information centres			✓		
51	local copy	remote version	027 General libraries, archives, information centres		✓			610
			610 Medical Sciences, Medicine				✓	
52	local copy	remote version	340 Law			✓		32?
			320 Political Science				✓	
53	local copy	remote version	025.5 Services to users (Operations of libraries, archives, information centres	✓				001.4
			380 Commerce, Communications, Transportation	✓				
54	local copy	remote version	004.1 General works on specific types of computers		✓			025.31
			027 General libraries, archives, information centres				✓	
55	local copy	remote version	340 Law	✓				332.4791
			630 Agriculture and Related Technologies	✓				
56	local copy	remote version	370 Education				✓	378
			004.6 Interfacing and communications (Computer science)		✓			
57	local copy	remote version	750 Painting and Paintings	✓				323.3383
			740 Drawing and Decorative Arts	✓				
58	local copy	remote version	370 Education				✓	378
			530 Physics				✓	
59	local copy	remote version	340 Law				✓	340.071
			370 Education				✓	
60	local copy	remote version	004.6 Interfacing and communications (Computer science)	✓				368
			005.3 Programs (Computer programs)	✓				
61	local copy	remote version	590 Animals	✓				025.21 (difficult from info.
62	local copy	remote version	820 English and Old English Literatures	✓				942.424
63	local copy	remote version	590 Animals				✓	599.5
			310 Collections of General Statistics	✓				

64	local copy	remote version	070.1 Documentary media, educational media, news media	✓				004.678
			004.6 Interfacing and communications (Computer science)				✓	
65	local copy	remote version	750 Painting and Paintings					332.6
			660 Chemical Engineering					
66	local copy	remote version	380 Commerce, Communications, Transportation		✓			260.029
			320 Political Science	✓				
67	local copy	remote version	004.7 Peripherals (Computer science)				✓	004.7
			410 Linguistics	✓				
68	local copy	remote version	650 Management and Auxiliary Services	✓				352.1409 4227
			640 Home Economics and Family Living	✓				
69	local copy	remote version	004.6 Interfacing and communications (Computer science)				✓	004.68
			005.1 Programming (Computer programming)			✓		
70	local copy	remote version	370 Education				✓	371.33
			520 Astronomy and Allied Sciences	✓				
71	local copy	remote version	370 Education				✓	378
72	local copy	remote version	320 Political Science		✓			305.90664
			305 Social Groups				✓	
73	local copy	remote version	660 Chemical Engineering	✓				004.6
			380 Commerce, Communications, Transportation		✓			
74	local copy	remote version	660 Chemical Engineering			✓		622.338
			650 Management and Auxiliary Services	✓				
75	local copy	remote version	027 General libraries, archives, information centres			✓		025.21
			025.5 Services to users (Operations of libraries, archives, information centres			✓		
76	local copy	remote version	790 Recreational and Performing Arts					378.1
			780 Music					
77	local copy	remote version	380 Commerce, Communications, Transportation			✓		338.7
78	local copy	remote version	650 Management and Auxiliary Services					can't open site
79	local copy	remote version	340 Law	✓				363.7
			350 Public Administration and Military Science	✓				

80	local copy	remote version	004.0151 Mathematical principles (Computer Systems)				✓	004.0151
81	local copy	remote version	005.7 Data in computer systems				✓	005.7
			004.1 General works on specific types of computers			✓		
82	local copy	remote version	004.6 Interfacing and communications (Computer science)	✓				025.21
			003.7 Kinds of Systems (Computer Systems)	✓				
83	local copy	remote version	780 Music				✓	780.7841835
			250 Christian Orders and Local Church	✓				
84	local copy	remote version	307 Communities		✓			361.2
			305 Social Groups	✓				
85	local copy	remote version	370 Education		✓			331.128
			340 Law	✓				
86	local copy	remote version	150 Psychology	✓				278.1
			005.1 Programming (Computer programming)	✓				
87	local copy	remote version	027 General libraries, archives, information centres			✓		335.4
			004.6 Interfacing and communications (Computer science)	✓				
88	local copy	remote version	004.1 General works on specific types of computers				✓	004
			005.4 Systems programming and programs (Computer programs)		✓			
89	local copy	remote version	790 Recreational and Performing Arts				✓	796 1
			710 Civic and landscape Art	✓				
90	local copy	remote version	650 Management and Auxiliary Services		✓			378.12
			330 Economics	✓				
91	local copy	remote version	750 Painting and Paintings	✓				332.6
			660 Chemical Engineering	✓				
92	local copy	remote version	070.1 Documentary media, educational media, news media	✓				551 6
			790 Recreational and Performing Arts	✓				
93	local copy	remote version	690 Buildings		✓			362.5
			320 Political Science		✓			
94	local copy	remote version	380 Commerce, Communications, Transportation				✓	388.342
95	local copy	remote version	620 Engineering and Allied Operations		✓			373 1
			003.7 Kinds of Systems (Computer Systems)		✓			

96	local copy	remote version	005.7 Data in computer systems	✓				025.21
			790 Recreational and Performing Arts	✓				
97	local copy	remote version	027 General libraries, archives, information centres				✓	025 81
			025.5 Services to users (Operations of libraries, archives, information centres				✓	
98	local copy	remote version	370 Education			✓		032
			350 Public Administration and Military Science	✓				
99	local copy	remote version	370 Education				✓	371.26
100	local copy	remote version	620 Engineering and Allied Operations	✓				323.64
			310 Collections of General Statistics	✓				
101	local copy	remote version	760 Graphic Arts Printmaking Prints				✓	760
102	local copy	remote version	660 Chemical Engineering	✓				361 7
			640 Home Economics and Family Living	✓				
103	local copy	remote version	670 Manufacturing	✓				387 736
			640 Home Economics and Family Living	✓				
104	local copy	remote version	610 Medical Sciences, Medicine				✓	616.951
			025.5 Services to users (Operations of libraries, archives, information centres	✓				
105	local copy	remote version	005.1 Programming (Computer programming)			✓		004.678
			004.6 Interfacing and communications (Computer science)				✓	
106	local copy	remote version	370 Education				✓	378
			690 Buildings	✓				
107	local copy	remote version	690 Buildings		✓			328.4791 (difficult from info.)
108	local copy	remote version	640 Home Economics and Family Living	✓				338.4791
			790 Recreational and Performing Arts	✓				
109	local copy	remote version	640 Home Economics and Family Living		✓			321.128
110	local copy	remote version	640 Home Economics and Family Living	✓				790.13 (difficult to specify)
			004.6 Interfacing and communications (Computer science)		✓			
111	local copy	remote version	380 Commerce, Communications, Transportation		✓			338.4791
			640 Home Economics and Family Living	✓				

112	local copy	remote version	070.1 Documentary media, educational media, news media	✓				363.123
			380 Commerce, Communications, Transportation		✓			
113	local copy	remote version	006.7 Multimedia (Special computer methods	✓				796.5
			006.6 Computer graphics	✓				
114	local copy	remote version	380 Commerce, Communications, Transportation	✓				306 72
			320 Political Science	✓				
115	local copy	remote version	620 Engineering and Allied Operations	✓				387.7
			790 Recreational and Performing Arts	✓				
116	local copy	remote version	530 Physics				✓	520
			520 Astronomy and Allied Sciences	✓				
117	local copy	remote version	790 Recreational and Performing Arts	✓				371 24
			780 Music	✓				
118	local copy	remote version	302 Social Interaction	✓				200 71
			120 Epistemology, Causation, Humankind	✓				
119	local copy	remote version	620 Engineering and Allied Operations	✓				910 (specifics by section)
			310 Collections of General Statistics	✓				
120	local copy	remote version	320 Political Science				✓	320
			490 Other Languages	✓				
121	local copy	remote version	530 Physics	✓				378.19
			005.7 Data in computer systems	✓				
122	local copy	remote version	350 Public Administration and Military Science	✓				610
			005.8 Data security	✓				
123	local copy	remote version	070.1 Documentary media, educational media, news media	✓				384.53
			780 Music	✓				
124	local copy	remote version	780 Music	✓				491.6
			070.1 Documentary media, educational media, news media	✓				
125	local copy	remote version	250 Christian Orders and Local Church		✓			726 =
			230 Christianity Christian Theology	✓				
126	local copy	remote version	320 Political Science				✓	324.214
			070.1 Documentary media, educational media, news media	✓				
127	local copy	remote version	380 Commerce, Communications, Transportation				✓	382.342
			590 Animals	✓				

128	local copy	remote version	005.7 Data in computer systems	/				384
			380 Commerce, Communications, Transportation				/	
129	local copy	remote version	790 Recreational and Performing Arts				✓	796.352
			780 Music	✓				
130	local copy	remote version	790 Recreational and Performing Arts			/		613.7
			380 Commerce, Communications, Transportation	✓				
131	local copy	remote version	590 Animals	✓				500
			004.6 Interfacing and communications (Computer science)	✓				
132	local copy	remote version	370 Education				✓	378.103
133	local copy	remote version	590 Animals	✓				328 4791
			780 Music	✓				
134	local copy	remote version	350 Public Administration and Military Science				✓	352.4183
			006.3 Artificial intelligence	✓				
135	local copy	remote version	790 Recreational and Performing Arts				✓	792
			004.1 General works on specific types of computers	✓				
136	local copy	remote version	004.6 Interfacing and communications (Computer science)	✓				370.7
			370 Education				✓	
137	local copy	remote version	006.6 Computer graphics		✓			700
			750 Painting and Paintings	✓				
138	local copy	remote version	350 Public Administration and Military Science	✓				372.216
139	local copy	remote version	004.6 Interfacing and communications (Computer science)	✓				942.982
			508 Natural history	✓				
140	local copy	remote version	570 Life Sciences, Biology		✓			025.21
			530 Physics		✓			
141	local copy	remote version	610 Medical Sciences, Medicine				✓	616.12302 5
			370 Education	✓				
142	local copy	remote version	380 Commerce, Communications, Transportation	✓				650.14
			310 Collections of General Statistics	✓				
143	local copy	remote version	790 Recreational and Performing Arts				✓	796 342
			340 Law	✓				

144	local copy	remote version	004.7 Peripherals (Computer science)	✓				946.8
			190 Modern Western Philosophy	✓				
145	local copy	remote version	660 Chemical Engineering	✓				362.728
			650 Management and Auxiliary Services	✓				
146	local copy	remote version	790 Recreational and Performing Arts	✓				282
147	local copy	remote version	640 Home Economics and Family Living	✓				328.4791
			790 Recreational and Performing Arts	✓				
148	local copy	remote version	380 Commerce, Communications, Transportation	✓				328.4791
149	local copy	remote version	780 Music				✓	780
			070.1 Documentary media, educational media, news media	✓				
150	local copy	remote version	380 Commerce, Communications, Transportation				✓	328.342
151	local copy	remote version	790 Recreational and Performing Arts				✓	796.334
			070.1 Documentary media, educational media, news media		✓			
152	local copy	remote version	670 Manufacturing	✓				650.14
			380 Commerce, Communications, Transportation	✓				
153	local copy	remote version	510 Mathematics				✓	516
154	local copy	remote version	004.6 Interfacing and communications (Computer science)			✓		005.72
			005.1 Programming (Computer programming)				✓	
155	local copy	remote version	370 Education				✓	378.124
156	local copy	remote version	370 Education				✓	378
			650 Management and Auxiliary Services	✓				
157	local copy	remote version	780 Music				✓	784.2
			790 Recreational and Performing Arts	✓				
158	local copy	remote version	380 Commerce, Communications, Transportation	✓				551.457 (need more info)
159	local copy	remote version	350 Public Administration and Military Science	✓				658.3124
			307 Communities	✓				

160	local copy	remote version	025.5 Services to users (Operations of libraries, archives, information centres	✓				378
			005.7 Data in computer systems	✓				
161	local copy	remote version	350 Public Administration and Military Science				✓	355.007
			740 Drawing and Decorative Arts	✓				
162	local copy	remote version	004.6 Interfacing and communications (Computer science)			✓		004.16
			380 Commerce, Communications, Transportation	✓				
163	local copy	remote version	004.0151 Mathematical principles (Computer Systems)	✓				331.128
164	local copy	remote version	027 General libraries, archives, information centres	✓				305.8
			690 Buildings	✓				
165	local copy	remote version	380 Commerce, Communications, Transportation	✓				942.982
			307 Communities	✓				
166	local copy	remote version	640 Home Economics and Family Living				✓	641.5
			025.5 Services to users (Operations of libraries, archives, information centres	✓				
167	local copy	remote version	070.1 Documentary media, educational media, news media				✓	076
			070.4 Journalism				✓	
168	local copy	remote version	650 Management and Auxiliary Services	✓				700.94227
169	local copy	remote version	650 Management and Auxiliary Services		✓			331.128
			750 Painting and Paintings	✓				
170	local copy	remote version	370 Education				✓	378
			650 Management and Auxiliary Services	✓				
171	local copy	remote version	370 Education				✓	378
			650 Management and Auxiliary Services	✓				
172	local copy	remote version	640 Home Economics and Family Living		✓			387.20423
			790 Recreational and Performing Arts			✓		
173	local copy	remote version	690 Buildings	✓				378.19
			380 Commerce, Communications, Transportation	✓				
174	local copy	remote version	690 Buildings		✓			645 3
			680 Manufacture for Specific uses	✓				
175	local copy	remote version	370 Education				✓	378.125
			004.6 Interfacing and communications (Computer science)		✓			

176	local copy	remote version	004.6 Interfacing and communications (Computer science)	✓				030
			005.7 Data in computer systems	✓				
177	local copy	remote version	005.7 Data in computer systems	✓				491.6
178	local copy	remote version	005.7 Data in computer systems				✓	005.7
			025.5 Services to users (Operations of libraries, archives, information centres	✓				
179	local copy	remote version	340 Law				✓	340.0711
			027 General libraries, archives, information centres	✓				
180	local copy	remote version	370 Education			✓		331.128
			650 Management and Auxiliary Services	✓				
181	local copy	remote version	790 Recreational and Performing Arts				✓	796.33463
			680 Manufacture for Specific uses	✓				
182	local copy	remote version	720 Architecture				✓	946.8
			708 Galleries, Museums, Private Collections				✓	
183	local copy	remote version	780 Music			✓		792.23
			070.1 Documentary media, educational media, news media	✓				
184	local copy	remote version	780 Music				✓	780.9417
			070.1 Documentary media, educational media, news media	✓				
185	local copy	remote version	370 Education				✓	378.19
			330 Economics	✓				
186	local copy	remote version	370 Education			✓		331.2592
			350 Public Administration and Military Science	✓				
187	local copy	remote version	027 General libraries, archives, information centres				✓	025.8
			025.8 Maintenance and preservation (Operations of libraries, archives, information centres				✓	
188	local copy	remote version	305 Social Groups		✓			615.1
			610 Medical Sciences, Medicine				✓	
189	local copy	remote version	190 Modern Western Philosophy	✓				181.4
			110 Metaphysics		✓			
190	local copy	remote version	370 Education				✓	372.216
191	local copy	remote version	380 Commerce, Communications, Transportation				✓	380 (based on available information)

192	local copy	remote version	670 Manufacturing	✓				616.853
			610 Medical Sciences, Medicine				✓	
193	local copy	remote version	004.6 Interfacing and communications (Computer science)	✓				328.7
			004.3 Processing modes (Computer science)	✓				
194	local copy	remote version	690 Buildings			✓		610.7
			022 Administration of the physical plant (Library and information sciences)	✓				
195	local copy	remote version	307 Communities		✓			796.5
			790 Recreational and Performing Arts				✓	
196	local copy	remote version	370 Education				✓	370.7
			021 Relationships of libraries, archives, information centres					
197	local copy	remote version	790 Recreational and Performing Arts				✓	796 93
			750 Painting and Paintings	✓				
198	local copy	remote version	340 Law		✓			378.194
			380 Commerce, Communications, Transportation	✓				
199	local copy	remote version	790 Recreational and Performing Arts				✓	794 8
200	local copy	remote version	650 Management and Auxiliary Services	✓				363.2
			380 Commerce, Communications, Transportation	✓				

APPENDIX J

Spreadsheet Combining Librarian Results.

The following pages show a spreadsheet combining the results of the three librarians. Each column of the spreadsheet represents the following:

A	Accession number
B	Automatically assigned classmark 1
C	Classmark 1 score
D	Automatically assigned classmark 2 (where applicable)
E	Classmark 2 score (where applicable)
F	Librarian 1 rating for classmark 1
G	Librarian 1 rating for classmark 2 (where applicable)
H	Librarian 2 rating for classmark 1
I	Librarian 2 rating for classmark 2 (where applicable)
J	Librarian 3 rating for classmark 1
K	Librarian 3 rating for classmark 2 (where applicable)
L	Manually assigned classmark from librarian 1
M	Manually assigned classmark from librarian 2
N	Manually assigned classmark from librarian 3
O	Total number of 4 ratings for classmark 1
P	Total number of 3 ratings for classmark 1
Q	Total number of 2 ratings for classmark 1
R	Total number of 1 ratings for classmark 1
S	Total number of 4 ratings for classmark 2 (where applicable)
T	Total number of 3 ratings for classmark 2 (where applicable)
U	Total number of 2 ratings for classmark 2 (where applicable)
V	Total number of 1 ratings for classmark 2 (where applicable)
W	Total number of manual classifications in common

A	B	C	D	E	F
1	370 Education	4.4			4
2	640 Home Economics and Family Living	6.6			1
3	004.6 Interfacing and Communications (Computer Science)	5.46	005.7 Data in Computer Systems	3.72	4
4	640 Home Economics and Family Living	2.34	790 Recreational and Performing Arts	1.24	4
5	320 Political Science	1.54	190 Modern Western Philosophy	1.32	4
6	070.1 Documentary Media, Educational Media, News Media	15.4	340 Law	8.8	3
7	690 Buildings	3.5	640 Home Economics and Family Living	2.5	4
8	640 Home Economics and Family Living	1.42	380 Commerce, Communications, Transportation	1.02	1
9	380 Commerce, Communications, Transportation	2.2			1
10	370 Education	4.4	304 Social Behaviour	4.13	4
11	510 Mathematics	3.8	150 Psychology	2.14	3
12	330 Economics	2.86	650 Management and Auxiliary Services	1.76	2
13	708 Galleries, Museums, Private Collections	1.76	069 Museology (Museum Science)	0.8	4
14	610 Medical Sciences, Medicine	8.14	370 Education	1.43	4
15	520 Astronomy and Allied Sciences	4			1
16	307 Communities	3.8	305 Social Groups	2.6	4
17	003.5 Theory of Communication and Control (Computer Science)	3.46			2
18	004.6 Interfacing and Communications (Computer Science)	16.1	680 Manufacture for Specific uses	3.6	4
19	370 Education	2.2	021 Relationships of Libraries, Archives, Information Centres	1.58	2
20	790 Recreational and Performing Arts	6.32	005.3 Programs (Computer Science)	2.96	4
21	370 Education	2.42	021 Relationships of Libraries, Archives, Information Centres	2.2	2
22	380 Commerce, Communications, Transportation	3.21			3
23	650 Management and Auxiliary Services	2	027 General Libraries, Archives, Information Centres	1.8	1
24	590 Animals	1.56	690 Buildings	1.48	1
25	340 Law	2.04			1
26	670 Manufacturing	1.32	005.7 Data in Computer Systems	0.72	1
27	610 Medical Sciences, Medicine	11.3	370 Education	3.4	1
28	005.7 Data in Computer Systems	5.4	005.1 Programming (Computer Programming)	4.2	3
29	070.1 Documentary Media, Educational Media, News Media	5.68	070.4 Journalism	5.46	3
30	027 General Libraries, Archives, Information Centres	12.56	690 Buildings	4.12	4
31	340 Law	4			3
32	370 Education	8.4	650 Management and Auxiliary Services	2.2	4
33	070.1 Documentary Media, Educational Media, News Media	1.46	350 Public Administration and Military Science	1.32	2

G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
	4		4		"378"	"378"	"378"	3	0	0	0					3
	2		1		"696.1"	"696.1"	"696.1"	0	0	1	2					3
3	3	4	3	4	"004.6"	"005.72"	"005.72"	1	2	0	0	2	1	0	0	2
3	4	3	4	3	"647.9442342"	"647"	"647.95"	3	0	0	0	0	3	0	0	3
3	4	3	4	3	"320"	"321"	"324"	3	0	0	0	0	3	0	0	3
1	4	2	1	1	"302.2"	"070.1"	"004.6"	1	1	0	1	0	0	1	2	0
2	4	4	1	4	"690"	"690"	"643.12"	2	0	0	1	2	0	1	0	2
3	2	4	1	3	"338.7"	"380.1"	"338.71"	0	0	1	2	1	2	0	0	2
	1		1		"378.11"	"378.11"	"378"	0	0	0	3					3
2	4	2	4	3	"372"	"372"	"370.115"	3	0	0	0	0	1	2	0	3
1	4	1	3	1	"507.11"	"507.11"	"500"	1	2	0	0	0	0	0	3	3
4	2	4	4	4	"650.14"	"650.14"	"657.8"	1	0	2	0	3	0	0	0	3
3	4	3	4	2	"708"	"708"	"708"	3	0	0	0	0	2	1	0	3
3	4	4	4	3	"610.730711"	"610.711"	"610.7"	3	0	0	0	1	2	0	0	3
	1		1		"338.7"	"600"	"338.926"	0	0	0	3					2
3	4	3	4	3	"307"	"307"	"307.34"	3	0	0	0	0	3	0	0	3
	4		4		"338.7072"	"003.5"	"003.5"	2	0	1	0					2
1	4	4	4	1	"004.6"	"004.6"	"004.6"	3	0	0	0	1	0	0	2	3
4	4	3	3	4	"021"	"370"	"021.65"	1	1	1	0	2	1	0	0	2
3	4	4	4	4	"790"	"005.3"	"790.1"	3	0	0	0	2	1	0	0	2
4	4	3	4	3	"021"	"374"	"371.9"	2	0	1	0	1	2	0	0	2
	1		1		"331.8811365"	"365"	"331.881"	0	1	0	2					2
4	3	3	3	4	"020.711"	"020"	"027.007"	0	2	0	1	2	0	0	1	3
2	2	3	1	1	"133.3337"	"133"	"133.3337"	0	0	1	2	0	1	1	1	3
	1		1		"338.4791099646"	"643"	"338.4791"	0	0	0	3					2
3	1	4	1	1	"700.285"	"005.7"	"700.9"	0	0	0	3	1	1	0	1	2
3	4	3	4	2	"570.79"	"610"	"610.7"	2	0	0	1	0	2	1	0	2
1	4	3	4	4	"570.79"	"005.7"	"660.6" ?	2	1	0	0	1	1	0	1	0
4	4	4	?	?	"070.4"	"070.1"	"005.7"	1	1	0	0	2	0	0	0	2
1	4	2	4	1	"027"	"027"	"027.2"	3	0	0	0	0	0	1	2	3
	4		4		"658.88"	"346"	"346.077"	2	1	0	0					2
1	4	3	4	2	"372.241"	"371.2"	"371.2"	3	0	0	0	0	1	1	1	3
1	2	1	1	1	"791.437"	"380.1"	"791.4375"	0	0	2	1	0	0	0	3	2

34	820 English and Old English Literatures	8.8	790 Recreational and Performing Arts	6.57	2
35	027 General Libraries, Archives, Information Centres	9	022 Administration of the Physical Plant (Library and Information Sciences)	1.7	1
36	790 Recreational and Performing Arts	10.12	005.8 Data Security	6.6	1
37	670 Manufacturing	2.236	380 Commerce, Communications, Transportation	2.1	1
38	790 Recreational and Performing Arts	12.36	004.6 Interfacing and Communications (Computer Science)	5.68	1
39	780 Music	3.82	005.1 Programming (Computer Programming)	1.32	4
40	027 General Libraries, Archives, Information Centres	2.5	420 English	1.5	2
41	790 Recreational and Performing Arts	4.4	780 Music	2.5	4
42	004.1 General Works on Specific Types of Computers	3.06	005.4 Systems Programming and Programs (Computer Programs)	1.86	1
43	790 Recreational and Performing Arts	3	780 Music	2.2	3
44	790 Recreational and Performing Arts	1.6			4
45	307 Communities	4.2	305 Social Groups	2.2	4
46	370 Education	1.54	350 Public Administration and Military Science	1.32	3
47	510 Mathematics	5.79	370 Education	3.74	4
48	025.5 Services to Users (Operations of Libraries, Archives and Information Centres)	4.36	027 General Libraries, Archives, Information Centres	3.5	1
49	070.1 Documentary Media, Educational Media, News Media	4.4			1
50	370 Education	6.12	027 General Libraries, Archives, Information Centres	3.26	4
51	027 General Libraries, Archives, Information Centres	4.44	610 Medical Sciences, Medicine	2.46	4
52	340 Law	4.41	320 Political Sciences	3.24	3
53	025.5 Services to Users (Operations of Libraries, Archives and Information Centres)	1.04	380 Commerce, Communications, Transportation	0.9	4
54	004.1 General Works on Specific Types of Computers	4.98	027 General Libraries, Archives, Information Centres	4.84	3
55	340 Law	2	630 Agriculture and Related Technologies	1.6	1
56	370 Education	10.6	004.6 Interfacing and Communications (Computer Science)	4.31	3
57	750 Painting and Paintings	6.8	740 Drawing and Decorative Arts	3.71	1
58	370 Education	1.72	530 Physics	1.46	3
59	340 Law	1.46	370 Education	1.378	4
60	004.6 Interfacing and Communications (Computer Science)	11.88	005.3 Programs (Computer Science)	5.68	1
61	590 Animals	6.3			1
62	820 English and Old English Literatures	7.93			1
63	590 Animals	9	310 Collections of General Statistics	3.2	4
64	070.1 Documentary Media, Educational Media, News Media	2.42	004.6 Interfacing and Communications (Computer Science)	1.72	4
65	750 Painting and Paintings	2.464	660 Chemical Engineering	1.74	1
66	380 Commerce, Communications, Transportation	2.2	320 Political Science	2.43	1
67	004.7 Peripherals	5.8	410 Linguistics	1.95	2

2	4	3	3	1	"381.45002"	"380.1"	"381.45002"	1	1	1	0	0	1	1	1	3
1	3	2	1	1	"378.161"	"378"	"378"	0	1	0	2	0	0	1	2	3
1	2	3	1	1	"352.669"	"355"	"352.379"	0	0	1	2	0	1	0	2	3
1	1	1	1	1	"150.92"	"004"	"650.14"	0	0	0	3	0	0	0	3	0
2	4	1	1	1	"338.4791"	"338.4791"	"338.4791"	1	0	0	2	0	0	1	2	3
1	4	1	4	1	"784.2"	"784"	"784.2"	3	0	0	0	0	0	0	3	3
1	4	1	3	1	"335.4092"	"335"	"335.5"	1	1	1	0	0	0	0	3	3
1	4	1	4	2	"793.932"	"790.1"	"793"	3	0	0	0	0	0	1	2	3
1	4	3	2	2	"331.128"	"004.1"	"331.128"	1	0	1	1	0	1	1	1	2
4	4	4	3	4	"780"	"780"	"780.905"	1	2	0	0	3	0	0	0	3
	4		4		"796.8152"	"796"	"796.8152"	3	0	0	0					3
3	4	4	4	2	"307"	"307"	"307"	3	0	0	0	1	1	1	0	3
1	4	1	4	2	"570.79"	"570.7"	"370.7"	2	1	0	0	0	0	1	2	2
3	4	4	4	3	"510.92"	"378"	"571"	3	0	0	0	1	2	0	0	0
1	4	3	1	1	"004.07114221"	"025.5"	"384"	1	0	0	2	0	1	0	2	0
	1		1		"914.604"	"647"	"338.4791"	0	0	0	3					0
1	4	3	4	3	"371.67"	"372"	"370.7"	3	0	0	0	0	2	0	1	3
2	4	1	2	4	"027"	"027"	"610"	2	0	1	0	1	0	1	1	2
4	3	4	3	4	"323.09951"	"327"	"323"	0	3	0	0	3	0	0	0	3
2	4	1	1	1	"025.5"	"025.5"	"001.4"	2	0	0	1	0	0	1	2	2
4	4	4	2	4	"027"	"025.5"	"025.31"	1	1	1	0	3	0	0	0	3
1	1	1	1	1	"914.28804"	"647"	"338.4791"	0	0	0	3	0	0	0	3	0
4	4	4	4	2	"004.6"	"004.6"	"378"	2	1	0	0	2	0	1	0	2
1	1	1	1	1	"914.604"	"647"	"333.3383"	0	0	0	3	0	0	0	3	0
4	4	4	4	4	"530.071142496"	"378"	"378"	2	1	0	0	3	0	0	0	3
3	4	4	4	4	"340.07114134"	"378"	"340.0711"	3	0	0	0	2	1	0	0	2
1	4	4	1	1	"368.0065"	"004.6"	"368"	1	0	0	2	1	0	0	2	2
	1		1		"025.04"	"005.741"	"025.31"?	0	0	0	3					2
	1		1		"790.2"	"011.7"	"942.424"	0	0	0	3					0
1	4	2	4	1	"599.509468"	"590"	"599.5"	3	0	0	0	0	0	1	2	3
2	4	4	1	4	"070.19"	"070.1"	"004.678"	2	0	0	1	2	0	1	0	2
1	1	1	?	?	"332.6"	"332.6"	"332.6"	0	0	0	2	0	0	0	2	3
1	4	1	2	1	"264.033"	"380.1"	"260.029"	1	0	1	1	0	0	0	3	2
1	4	1	4	1	"704.9 ?"	"004.7"	"004.7"	2	0	1	0	0	0	0	3	2

68	650 Management and Auxiliary Services	4.7	640 Home Economics and Family Living	2.36	1
69	004.6 Interfacing and Communications (Computer Science)	7.8	005.1 Programming (Computer Programming)	7.56	4
70	370 Education	5.12	520 Astronomy and Allied Sciences	2.8	4
71	370 Education	2.1			4
72	320 Political Science	8.8	305 Social Groups	7.76	2
73	660 Chemical Engineering	1.98	380 Commerce, Communications, Transportation	0.97	1
74	660 Chemical Engineering	3.88	650 Management and Auxiliary Services	2.2	2
75	027 General Libraries, Archives, Information Centres	8.56	025.5 Services to Users (Operations of Libraries, Archives, Information Centres)	6.1	3
76	790 Recreational and Performing Arts	3.5	780 Music	2	2
77	380 Commerce, Communications, Transportation	8			2
78	650 Management and Auxiliary Services	2.27			?
79	340 Law	6.6	350 Public Administration and Military Science	6.2	1
80	004.0151 Mathematical Principles (Computer Science)	2.3			3
81	005.7 Data in Computer Systems	9.92	004.1 General Works on Specific Types of Computers	8.14	4
82	004.6 Interfacing and Communications (Computer Science)	1.02	003.7 Kinds of Systems (Computer Systems)	2.2	4
83	780 Music	3	250 Christian Orders and Local Church	1.02	3
84	307 Communities	2.6	305 Social Groups	2.42	4
85	370 Education	2.34	340 Law	1.06	2
86	150 Psychology	9.8	005.1 Programming (Computer Programming)	6	2
87	027 General Libraries, Archives, Information Centres	1.4	004.6 Interfacing and Communications (Computer Science)	1	2
88	004.1 General Works on Specific Types of Computers	3.52	005.4 Systems Programming and Programs (Computer Programs)	3.3	3
89	790 Recreational and Performing Arts	2.94	710 Civic and Landscape Art	1.72	4
90	650 Management and Auxiliary Services	3.52	330 Economics	2.42	1
91	750 Painting and Paintings	2.464	660 Chemical Engineering	1.543	1
92	070.1 Documentary Media, Educational Media, News Media	7.967	790 Recreational and Performing Arts	2.45	1
93	690 Buildings	10.6	320 Political Science	8.4	3
94	380 Commerce, Communications, Transportation	4			4
95	620 Engineering and Allied Operations	11.1	003.7 Kinds of Systems (Computer Systems)	4.04	4
96	005.7 Data in Computer Systems	3.88	790 Recreational and Performing Arts	1.1	2
97	027 General Libraries, Archives, Information Centres	3.8	025.5 Services to Users (Operations of Libraries, Archives, Information Centres)	2.2	3
98	370 Education	4.4	350 Public Administration and Military Science	2.2	2
99	370 Education	8			4
100	620 Engineering and Allied Operations	1.6	310 Collections of General Statistics	0.72	1
101	760 Graphic Arts Printmaking Prints	4			4

1	1	1	1	1	"790.079"	"011.7"	"352.14094227"	0	0	0	3	0	0	0	3	0
3	4	3	4	3	"004.6"	"004.6"	"004.68"	3	0	0	0	0	3	0	0	3
1	4	1	4	1	"370.285"	"370"	"371.33"	3	0	0	0	0	0	0	3	3
	4		4		"370.285"	"378"	"378"	3	0	0	0					3
3	?	4	2	4	"306.766"	"305"	"305.90664"	0	0	2	0	2	1	0	0	3
1	1	4	1	2	"004.6"	"004.6"	"004.6"	0	0	0	3	1	0	1	1	3
1	4	3	3	1	"354.44"	"660"	"622.338"	1	1	1	0	0	1	0	2	0
4	4	4	3	3	"025.5"	"025.5"	"025.31"	1	2	0	0	2	1	0	0	3
2	1	1	?	?	"378.166"	"378"	"378.1"	0	0	1	1	0	0	1	1	3
	4		3		"338.7"	"380.1"	"338.7"	1	1	1	0					2
	1		?		?	"796"	?	0	0	0	1					0
1	4	4	1	1	"333.7"	"344"	"363.7"	1	0	0	2	1	0	0	2	0
	3		4		"510"	"510"	"004.0151"	1	2	0	0					2
3	4	4	4	3	"005.7"	"005.7"	"005.7"	3	0	0	0	1	2	0	0	3
3	4	1	1	1	"004.6"	"004.6"	"025.31"	2	0	0	1	0	1	0	2	2
1	4	1	4	1	"621.3893"	"780"	"780.7841835"	2	1	0	0	0	0	0	3	2
3	4	3	2	1	"307"	"307"	"361.2"	2	0	1	0	0	2	0	1	2
3	4	1	2	1	"331.124"	"331"	"331.128"	1	0	2	0	0	0	0	3	3
3	4	4	1	1	"006.30711"	"378"	"378.1"	1	0	1	1	1	1	0	1	2
2	4	3	3	1	"335.4"	"335"	"335.4"	1	1	1	0	0	1	1	1	3
4	4	4	4	2	"005.4"	"004.1"	"004"	2	1	0	0	2	0	1	0	2
1	4	1	4	1	"793.92"	"796"	"796.1"	3	0	0	0	0	0	0	3	3
1	4	1	2	1	"378.1"	"658.3"	"378.12"	1	0	1	1	0	0	0	3	2
1	1	1	1	1	"332.6"	"332.6"	"332.6"	0	0	0	3	0	0	0	3	3
1	4	1	1	1	"551.654167"	"070.1"	"551.6"	1	0	0	2	0	0	0	3	2
4	2	3	2	2	"320.6"	"363.7"	"363.5"	0	1	2	0	1	1	1	0	2
	4		4		"388.4132"	"380.1"	"388.342"	3	0	0	0					3
1	4	1	2	2	"629.10711"	"378"	"378.1"	2	0	1	0	0	0	1	2	2
4	4	3	1	1	"790"	"005.7"	"025.31"	1	0	1	1	1	1	0	1	0
4	4	4	4	4	"025.5"	"025.5"	"025.31"	2	1	0	0	3	0	0	0	3
1	4	1	3	1	"025.5"	"025.5"	"032"	1	1	1	0	0	0	0	3	2
	4		4		"371.26"	"370"	"371.26"	3	0	0	0					3
1	1	1	1	1	"323.64"	"323"	"323.64"	0	0	0	3	0	0	0	3	3
	4		4		"760"	"760"	"760"	3	0	0	0					3

102	660 Chemical Engineering	3.6	640 Home Economics and Family Living	2.2	1
103	670 Manufacturing	5.98	640 Home Economics and Family Living	1.73	1
104	610 Medical Sciences, Medicine	3.74	025.5 Services to Users (Operations of Libraries, Archives, Information Centres	2.64	4
105	005.1 Programming (Computer Programming)	2.8	004.6 Interfacing and Communications (Computer Science)	2.8	2
106	370 Education	1.34	690 Buildings	0.6	3
107	690 Buildings	6			4
108	640 Home Economics and Family Living	5.9	790 Recreational and Performing Arts	1.76	1
109	640 Home Economics and Family Living	8.4			3
110	640 Home Economics and Family Living	17.8	004.6 Interfacing and Communications (Computer Science)	6.6	1
111	380 Commerce, Communications, Transportation	9.54	640 Home Economics and Family Living	4.34	4
112	070.1 Documentary Media, Educational Media, News Media	4.14	380 Commerce, Communications, Transportation	3.08	1
113	006.7 Multimedia (Special Computer Methods)	3.52	006.6 Computer Graphics	2.6	1
114	380 Commerce, Communications, Transportation	4	320 Political Science	2.2	1
115	620 Engineering and Allied Operations	4.4	790 Recreational and Performing Arts	1.7	1
116	530 Physics	14.32	520 Astronomy and Allied Sciences	4.2	4
117	790 Recreational and Performing Arts	3.1	780 Music	2.7	3
118	302 Social Interaction	13	120 Epistemology, Causation, Humankind	2.27	1
119	620 Engineering and Allied Operations	6.6	310 Collections of General Statistics	5.47	1
120	320 Political Science	5.6	490 Other Languages	4	4
121	530 Physics	11.2	005.7 Data in Computer Systems	7.8	3
122	350 Public Administration and Military Science	3.5	005.8 Data Security	1.5	2
123	070.1 Documentary Media, Educational Media, News Media	2.178	780 Music	1.782	2
124	780 Music	2.04	070.1 Documentary Media, Educational Media, News Media	1.1	4
125	250 Christian Orders and Local Church	3.24	230 Christianity Christian Theology	1.82	2
126	320 Political Science	1.98	070.1 Documentary Media, Educational Media, News Media	0.6	4
127	380 Commerce, Communications, Transportation	2.02	590 Animals	1.3	4
128	005.7 Data in Computer Systems	2.32	380 Commerce, Communications, Transportation	1.1	2
129	790 Recreational and Performing Arts	7.4	780 Music	2.37	4
130	790 Recreational and Performing Arts	4.36	380 Commerce, Communications, Transportation	3.973	4
131	590 Animals	21.42	004.6 Interfacing and Communications (Computer Science)	4.2	4
132	370 Education	6.2			4
133	590 Animals	1.6	780 Music	1.28	1
134	350 Public Administration and Military Science	3	006.3 Artificial Intelligence	2.5	2
135	790 Recreational and Performing Arts	5.18	004.1 General Works on Specific Types of Computers	1.68	2

1	1	3	1	1	"338.911"	"761.7"	"361.7"	0	0	0	3	0	1	0	2	0
1	2	1	1	1	"025.520285"	"387"	"387.736"	0	0	1	2	0	0	0	3	2
1	4	3	4	1	"610"	"610"	"616.951"	3	0	0	0	0	1	0	2	3
2	3	4	3	4	"745.40285"	"004.6"	"004.678"	0	2	1	0	2	0	1	0	2
1	4	1	4	1	"004.071142293"	"378"	"378"	2	1	0	0	0	0	0	3	2
	2		2		"690"	"647"	"338.4791"	1	0	2	0					0
1	4	4	1	1	"914.235"	"647"	"338.4791"	1	0	0	2	1	0	0	2	0
	?		2		"388.044"	"388"	"331.128"	0	1	1	0					2
4	4	4	1	2	"004.6"	"004.6"	"790.13"	1	0	0	2	2	0	1	0	2
1	3	3	2	1	"388.413214"	"338.4791"	"338.4791"	1	1	1	0	0	1	0	2	2
4	4	4	1	2	"387.542"	"070.1"	"363.123"	1	0	0	2	2	0	1	0	0
1	2	2	1	1	"381.142"	"380.1"	"796.5"	0	0	1	2	0	0	1	2	2
1	4	1	1	1	"646.770285"	"306"	"306.73"	1	0	0	2	0	0	0	3	2
4	1	4	1	1	"796.79"	"790"	"387.7"	0	0	0	3	2	0	0	1	2
1	4	4	4	1	"530"	"530"	"530"	3	0	0	0	1	0	0	2	3
3	2	1	1	1	"371.242"	"370"	"371.24"	0	1	1	1	0	1	0	2	2
1	1	3	1	1	"200.71"	"374"	"200.71"	0	0	0	3	0	1	0	2	2
1	1	3	1	1	"910.202"	"910"	"910"	0	0	0	3	0	1	0	2	3
1	4	1	4	1	"320.71"	"374"	"320"	3	0	0	0	0	0	0	3	2
1	4	2	1	1	"507.1"	"378"	"378.19"	1	1	0	1	0	0	1	2	2
2	4	1	1	1	"507.1"	"507"	"610"	1	0	1	1	0	0	1	2	2
1	4	2	1	1	"384.50285"	"070.1"	"384.53"	1	0	1	1	0	0	1	2	2
1	4	4	1	1	"780.79"	"306"	"491.6"	2	0	0	1	1	0	0	2	0
2	4	4	2	1	"726.5"	"250"	"726.5"	1	0	2	0	1	0	1	1	2
1	4	4	4	1	"324.24104"	"320"	"324.214"	3	0	0	0	1	0	0	2	2
1	4	1	4	1	"388.342"	"380"	"388.342"	3	0	0	0	0	0	0	3	3
4	4	4	1	4	"384.6"	"384"	"384"	1	0	1	1	3	0	0	0	3
1	4	2	4	1	"796.352068"	"790"	"796.352"	3	0	0	0	0	0	1	2	3
1	4	4	3	1	"796"	"790"	"613.7"	2	1	0	0	1	0	0	2	2
1	1	3	1	1	"808.838762"	"070.1"	"500"	1	0	0	2	0	1	0	2	0
	4		4		"378.101"	"378"	"378.103"	3	0	0	0					3
1	1	1	1	1	"914.129"	"338.4791"	"338.4791"	0	0	0	3	0	0	0	3	2
1	4	1	4	1	"387.73"	"387"	"358.4183"	2	0	1	0	0	0	0	3	2
1	4	4	4	1	"381.142"	"790"	"793"	2	0	1	0	1	0	0	2	2

136	004.6 Interfacing and Communications (Computer Science)	4.18	370 Education	1.76	1
137	006.6 Computer Graphics	6.6	750 Painting and Paintings	4.4	4
138	350 Public Administration and Military Science	2.82			1
139	004.6 Interfacing and Communications (Computer Science)	6.92	508 natural History	3.5	1
140	570 Life Sciences, Biology	7.5	530 Physics	3.56	3
141	610 Medical Sciences, Medicine	6.32	370 Education	1.79	4
142	380 Commerce, Communications, Transportation	5.8	310 Collections of General Statistics	3	1
143	790 Recreational and Performing Arts	12.04	340 Law	4	4
144	004.7 Peripherals (Computer Science)	10.74	190 Modern Western Philosophy	3.32	1
145	660 Chemical Engineering	8	650 Management and Auxiliary Services	6.8	1
146	790 Recreational and Performing Arts	4			?
147	640 Home Economics and Family Living	8.28	790 Recreational and Performing Arts	5	1
148	380 Commerce, Communications, Transportation	2.2			1
149	780 Music	13.28	070.1 Documentary Media, Educational Media, News Media	8.4	4
150	380 Commerce, Communications, Transportation	3.6			4
151	790 Recreational and Performing Arts	10.84	070.1 Documentary Media, Educational Media, News Media	1.54	4
152	670 Manufacturing	2.2	380 Commerce, Communications, Transportation	0.82	1
153	510 Mathematics	4			4
154	004.6 Interfacing and Communications (Computer Science)	2.2	005.1 Programming (Computer Programming)	1.5	3
155	370 Education	6			4
156	370 Education	3.1	650 Management and Auxiliary Services	2.16	3
157	780 Music	4.88	790 Recreational and Performing Arts	3.08	4
158	380 Commerce, Communications, Transportation	4.4			2
159	350 Public Administration and Military Science	7.9	307 Communities	6.18	2
160	025.5 Services to Users (Operations of Libraries, Archives, Information Centres)	9.04	005.7 Data in Computer Systems	8.74	4
161	350 Public Administration and Military Science	1.8	740 Drawing and Decorative	1.1	1
162	004.6 Interfacing and Communications (Computer Science)	6.4	380 Commerce, Communications, Transportation	1.42	4
163	004.0151 Mathematical Principles (Computer Science)	4.8			1
164	027 General Libraries, Archives, Information Centres	2.56	690 Buildings	3.95	4
165	380 Commerce, Communications, Transportation	9	307 Communities	4	2
166	640 Home Economics and Family Living	14.84	025.5 Services to Users (Operations of Libraries, Archives, Information Centres)	4.94	4
167	070.1 Documentary Media, Educational Media, News Media	4.5	070.4 Journalism	2.52	3
168	650 Management and Auxiliary Services	4.4			1
169	650 Management and Auxiliary Services	12.6	750 Painting and Paintings	8.8	1

4	4	4	1	4	"371.33"	"370"	"370.7"	1	0	0	2	3	0	0	0	3
3	4	4	2	1	"006.6"	"006.6"	"700"	2	0	1	0	1	1	0	1	2
	1		1		"372.216"	"370"	"372.216"	0	0	0	3					3
2	1	4	1	1	"942.982"	"508"	"942.982"	0	0	0	3	1	0	1	1	2
3	4	4	2	2	"501"	"570"	"025.31"	1	1	1	0	1	1	1	0	0
1	4	2	4	1	"616.12"	"610"	"616.123025"	3	0	0	0	0	0	1	2	3
1	?	1	1	1	"004.092"	"001.4" ?	"650.14"	0	0	0	2	0	0	0	3	0
1	4	1	4	1	"796.3424258"	"796"	"976.342"	3	0	0	0	0	0	0	3	2
1	2	1	1	1	"914.6804"	"946"	"946.8"	0	0	1	2	0	0	0	3	2
1	1	1	1	1	"363.728"	"620"	"363.728"	0	0	0	3	0	0	0	3	2
	4		1		?	"796"	"282"	1	0	0	1					0
1	2	4	1	1	"914.235"	"647"	"338.4791"	0	0	1	2	1	2	2	2	0
	4		1		"919.49504"	"338.4791"	"338.4791"	1	0	0	2					2
1	4	1	4	1	"781.642"	"780"	"780"	3	0	0	0	0	0	0	3	3
	4		4		"388.342"	"780.1"	"338.342"	3	0	0	0					2
3	4	4	4	2	"796"	"790"	"796.334"	3	0	0	0	1	1	1	0	3
1	1	1	1	1	"004.092"	"378"	"650.14"	0	0	0	3	3	3	3	3	0
	4		4		"516"	"510"	"516"	3	0	0	0					3
4	3	4	3	4	"005.3"	"005.1"	"005.72"	0	3	0	0	3	0	0	0	3
	4		4		"370.113"	"378"	"378.124"	3	0	0	0					3
1	4	3	4	1	"004.0711"	"378"	"378"	2	1	0	0	0	1	0	2	2
2	4	4	4	1	"784.2"	"780"	"784.2"	3	0	0	0	1	0	1	1	3
	1		1		"333.917"	"551"	"551.457"	0	0	1	2					2
2	3	4	1	1	"361.77"	"360"	"658.3124"	0	1	1	1	1	0	1	1	2
2	4	4	1	1	"025.5"	"025.5"	"378"	2	0	0	1	1	0	1	1	2
1	4	1	4	1	"070.1"	"355"	"355.007"	2	0	0	1	0	0	0	3	2
1	4	4	3	1	"004.6"	"004.6"	"004.16"	2	1	0	0	1	0	0	2	3
	3		1		"331.702"	"331"	"331.128"	0	1	0	2					3
1	4	1	1	1	"027"	"027"	"305.8"	2	0	0	1	0	0	0	3	2
1	2	4	1	1	"359.4"	"942"	"942.982"	0	0	2	1	1	0	0	2	2
1	4	2	4	1	"641.5"	"640"	"641.5"	3	0	0	0	0	0	1	2	3
3	4	4	4	4	"914.68"	"070.1"	"076"	2	1	0	0	2	1	0	0	2
	1		1		"709.4227"	"910"	"700.94227"	0	0	0	3					2
1	3	1	2	1	"331.124"	"331"	"331.128"	0	1	1	1	0	0	0	3	3

170	370 Education	5.8	650 Management and Auxiliary Services	4	1
171	370 Education	4.5	650 Management and Auxiliary Services	3	4
172	640 Home Economics and Family Living	2.2	790 Recreational and Performing Arts	1.64	1
173	690 Buildings	3	380 Commerce, Communications, Transportation	2.97	1
174	690 Buildings	4.87	680 Manufacture for Specific Uses	2.25	2
175	370 Education	4.18	004.6 Interfacing and Communications (Computer Science)	4	4
176	004.6 Interfacing and Communications (Computer Science)	1.98	005.7 Data in Computer Systems	1.54	2
177	005.7 Data in Computer Systems	1.06			2
178	005.7 Data in Computer Systems	6.26	025.5 Services to Users (Operations of Libraries, Archives, Information Centres)	3.4	4
179	340 Law	4.24	027 General Libraries, Archives, Information Centres	0.92	4
180	370 Education	6.68	650 Management and Auxiliary Services	1.89	3
181	790 Recreational and Performing Arts	4.4	680 Manufacture for Specific Uses	2.31	4
182	720 Architecture	6.6	708 Galleries, Museums, Private Collections	3.84	2
183	780 Music	3.6	070.1 Documentary Media, Educational Media, News Media	1.8	1
184	780 Music	7.58	070.1 Documentary Media, Educational Media, News Media	1.06	4
185	370 Education	2.56	330 Economics	0.4	4
186	370 Education	2.2	350 Public Administration and Military Science	1.63	2
187	027 General Libraries, Archives, Information Centres	9.64	025.8 Maintenance and Preservation (Operations of Libraries, Archives, Information Centres)	2.42	4
188	305 Social Groups	9.3	610 Medical Sciences, Medicine	5.7	1
189	190 Modern Western Philosophy	1.88	110 Metaphysics	1.22	3
190	370 Education	8.8			4
191	380 Commerce, Communications, Transportation	1.1			1
192	670 Manufacturing	6.27	610 Medical Sciences, Medicine	5.98	1
193	004.6 Interfacing and Communications (Computer Science)	2.08	004.3 Processing Modes(Computer Science)	1.42	4
194	690 Buildings	4.2	022 Administration of the Physical Plant (Library and Information Sciences)	3.6	3
195	307 Communities	1.9	790 Recreational and Performing Arts	1.1	2
196	370 Education	1.76	021 Relationships of Libraries, Archives, Information Centres	1.54	1
197	790 Recreational and Performing Arts	3.72	750 Painting and Paintings	2.88	3
198	340 Law	6	380 Commerce, Communications, Transportation	3	1
199	790 Recreational and Performing Arts	1.72			4
200	650 Management and Auxiliary Services	4.14	380 Commerce, Communications, Transportation	1.94	1

2	4	2	4	1	"363.20715"	"378"	"378"	2	0	0	1	0	0	2	1	2
1	4	1	4	1	"378.199"	"378"	"378"	3	0	0	0	0	0	0	3	3
3	3	3	2	3	"387.2"	"380.1"	"387.20423"	0	1	1	1	0	3	0	0	3
1	1	1	1	1	"378.19"	"378"	"378.19"	0	0	0	3	0	0	0	3	3
3	1	4	2	1	"645.3"	"680"	"645.32	0	0	2	1	1	1	0	1	2
3	4	4	4	2	"371.334"	"004.6"	"378.125"	3	0	0	0	1	1	1	0	2
2	4	3	1	1	"028.7"	"004.6"	"030"	1	0	1	1	0	1	1	1	0
	4		1		"028.7"	"005.7"	"491.6"	1	0	1	1					0
3	4	2	4	1	"005.741"	"005.7"	"005.7"	3	0	0	0	0	1	1	1	3
1	4	2	4	1	"346.048071"	"378"	"340.0711"	3	0	0	0	0	0	1	2	2
1	4	2	3	1	"331.124"	"370"	"331.128"	1	2	0	0	0	0	1	2	2
1	4	1	4	1	"796.33"	"796"	"796.33463"	3	0	0	0	0	0	0	3	3
3	3	3	4	4	"709.468"	"700"	"946.8"	1	1	1	0	1	2	0	0	2
2	2	2	3	1	"791.447"	"790"	"792.23"	0	1	1	1	0	0	2	1	3
2	4	3	4	1	"781.63"	"070.1"	"780.9417"	3	0	0	0	0	1	1	1	2
3	4	3	4	1	"378.19"	"378"	"378.19"	3	0	0	0	0	2	0	1	3
1	4	1	3	1	"646.7240715"	"374"	"331.2592"	1	1	1	0	0	0	0	3	0
3	4	4	4	4	"027"	"025.8"	"025.8"	3	0	0	0	2	1	0	0	3
4	2	4	2	4	"616.8527061"	"610"	"615.1"	0	0	2	1	3	0	0	0	3
3	4	2	1	2	"291.436"	"190"	"181.4"	1	1	0	1	0	1	2	0	0
	4		4		"372.9"	"372"	"372.216"	3	0	0	0					3
	3		4		"658.404 ?"	"658.2"	"380"	1	1	0	1					2
4	1	4	1	4	"616.853"	"610"	"616.853"	0	0	0	3	3	0	0	0	3
3	4	2	1	1	"004.6"	"004.6"	"338.7"	2	0	0	1	0	1	1	1	2
1	4	1	3	1	"610.71141443"	"378"	"610.7"	1	2	0	0	0	0	0	3	2
3	4	2	2	4	"361.77"	"360"	"796.5"	1	0	2	0	1	1	1	0	2
4	4	4	4	?	"025.5"	"370"	"370.7"	2	0	0	1	2	0	0	0	2
1	4	1	4	1	"914.404"	"647"	"796.93"	2	1	0	0	0	0	0	3	0
1	1	1	2	1	"378.19"	"378"	"378.194"	0	0	1	2	0	0	0	3	3
	4		4		"793.932"	"290"	"794.8"	3	0	0	0					2
1	2	1	1	1	"363.2094227"	"910"	"363.2"	0	0	1	2	0	0	0	3	2

APPENDIX K
The Wolverhampton Core RDF Schema

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
   xmlns:rdfs="http://www.w3.org/TR/WD-rdf-schema#">

<rdf:Description ID="Accession_no">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Identifier"/>
  <rdfs:label>Accession_no</rdfs:label>
  <rdfs:comment>A unique number assigned by the automatic classifier
  that uniquely identifies this resource.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Title">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Title"/>
  <rdfs:label>Title</rdfs:label>
  <rdfs:comment>The title of the resource taken from the HTML TITLE element.
  </rdfs:comment>
</rdf:Description>

<rdf:Description ID="Abstract">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Description"/>
  <rdfs:label>Abstract</rdfs:label>
  <rdfs:comment>This is the first 25 words taken from the BODY of the HTML
  page, or, if present, text taken from the description HTML META tag.
  </rdfs:comment>
</rdf:Description>

<rdf:Description ID="Keyword">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Subject"/>
  <rdfs:label>Keyword</rdfs:label>
  <rdfs:comment> This is a keyword from the document that matched a keyword
  in an appropriate DDC class representative. A number of keywords will
  normally appear in an RDF Bag container.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Classmark">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Subject"/>
  <rdfs:label>Classmark</rdfs:label>
  <rdfs:comment>This is a DDC classmark that has been assigned to the document
  as a result of the automatic classification process. Often two appropriate
  classmarks will be shown in an RDF sequence - the highest scoring one
  appearing first.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Word_count">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:label>Word_count</rdfs:label>
  <rdfs:comment>This is the number of individual words found in the
  resource.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Classification_date">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:label>Classification date</rdfs:label>
  <rdfs:comment>The date on which the resource was classified.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Last_modified">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Date"/>
  <rdfs:label>Last_modified</rdfs:label>
  <rdfs:comment>The date on which the resource was last modified
  when it was classified.</rdfs:comment>
</rdf:Description>

</rdf:RDF>
```

APPENDIX L
Automatically Generated RDF.

Automatically generated RDF for a selection of URLs taken from those used for the evaluation experiment in chapter 4.

L.1 University of Wolverhampton For Students - http://www.wlv.ac.uk/university/for.students.html

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://www.scit.wlv.ac.uk/~ex1253/wc/schema">
   <rdf:Description about ="http://www.wlv.ac.uk/university/for.students.html">
      <wc:Accession_no>0</wc:Accession_no>
      <wc:Title>University of Wolverhampton For Students</wc:Title>
      <wc:Abstract>For Students indicates an external site University String Orchestra
         Players Wanted Answer this questionnaire and you may win a PC Academic
         Schools Access maps Accommodation</wc:Abstract>
      <wc:Keyword>
         <rdf:Bag>
            <rdf:li>library</rdf:li>
            <rdf:li>pc</rdf:li>
            <rdf:li>television</rdf:li>
            <rdf:li>tv</rdf:li>
            <rdf:li>internet</rdf:li>
            <rdf:li>control</rdf:li>
            <rdf:li>email</rdf:li>
            <rdf:li>education</rdf:li>
            <rdf:li>school</rdf:li>
            <rdf:li>copyright</rdf:li>
            <rdf:li>service</rdf:li>
            <rdf:li>services</rdf:li>
            <rdf:li>access</rdf:li>
            <rdf:li>university</rdf:li>
            <rdf:li>chaplaincy</rdf:li>
            <rdf:li>christian</rdf:li>
            <rdf:li>telephone</rdf:li>
            <rdf:li>schools</rdf:li>
            <rdf:li>law</rdf:li>
            <rdf:li>regulations</rdf:li>
            <rdf:li>energy</rdf:li>
            <rdf:li>higher</rdf:li>
            <rdf:li>orchestra</rdf:li>
         </rdf:Bag>
      </wc:Keyword>
      <wc:Classmark>
         <rdf:Seq>
            <rdf:li>025.5   Services to users (Operations of libraries, archives,
               information centres</rdf:li>
            <rdf:li>027     General libraries, archives, information centres</rdf:li>
         </rdf:Seq>
      </wc:Classmark>
      <wc:Word_count>249</wc:Word_count>
      <wc:Classification_date>24-Sep-99 5:32:21 PM</wc:Classification_date>
      <wc:Last_modified>24-Sep-99 7:49:00 AM</wc:Last_modified>
   </rdf:Description>
</rdf:RDF>
```

L.2 Museum of Garden History St Mary at Lambeth South London - http://www.speel.demon.co.uk/other/gardhist.htm

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://www.scit.wlv.ac.uk/~ex1253/wc/schema">
   <rdf:Description about ="http://www.speel.demon.co.uk/other/gardhist.htm">
      <wc:Accession_no>0</wc:Accession_no>
      <wc:Title>Museum of Garden History St Mary at Lambeth South London</wc:Title>
      <wc:Abstract>Notes art in the collection of the Museum of Garden History
           Lambeth England</wc:Abstract>
      <wc:Keyword>
         <rdf:Bag>
            <rdf:li>museum</rdf:li>
            <rdf:li>church</rdf:li>
            <rdf:li>crucifixion</rdf:li>
            <rdf:li>war</rdf:li>
            <rdf:li>school</rdf:li>
            <rdf:li>dog</rdf:li>
            <rdf:li>horses</rdf:li>
            <rdf:li>gardening</rdf:li>
            <rdf:li>history</rdf:li>
            <rdf:li>buildings</rdf:li>
            <rdf:li>building</rdf:li>
            <rdf:li>material</rdf:li>
            <rdf:li>oil</rdf:li>
            <rdf:li>plant</rdf:li>
            <rdf:li>garden</rdf:li>
            <rdf:li>bridge</rdf:li>
            <rdf:li>human</rdf:li>
            <rdf:li>family</rdf:li>
            <rdf:li>windows</rdf:li>
            <rdf:li>window</rdf:li>
            <rdf:li>painting</rdf:li>
            <rdf:li>painter</rdf:li>
            <rdf:li>painters</rdf:li>
            <rdf:li>art</rdf:li>
            <rdf:li>artists</rdf:li>
            <rdf:li>gallery</rdf:li>
            <rdf:li>collection</rdf:li>
            <rdf:li>glass</rdf:li>
            <rdf:li>movement</rdf:li>
            <rdf:li>house</rdf:li>
         </rdf:Bag>
      </wc:Keyword>
      <wc:Classmark>
         <rdf:Seq>
            <rdf:li>630   Agriculture and Related Technologies</rdf:li>
            <rdf:li>708   Galleries, Museums, Private Collections</rdf:li>
         </rdf:Seq>
      </wc:Classmark>
      <wc:Word_count>649</wc:Word_count>
      <wc:Classification_date>24-Sep-99 5:35:08 PM</wc:Classification_date>
      <wc:Last_modified>10-Jul-99 10:38:19 PM</wc:Last_modified>
   </rdf:Description>
</rdf:RDF>
```

L.3 The Gardens - http://www.bham-bot-gdns.demon.co.uk/gardens.html

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://www.scit.wlv.ac.uk/-ex1253/wc/schema">
   <rdf:Description about ="http://www.bham-bot-gdns.demon.co.uk/gardens.html">
      <wc:Accession_no>0</wc:Accession_no>
      <wc:Title>The Gardens</wc:Title>
      <wc:Abstract>The Birmingham Botanical Gardens were opened in 1832 They were
         designed by J C Loudon a leading garden planner horticultural journalist and
         publisher Apart from</wc:Abstract>
      <wc:Keyword>
         <rdf:Bag>
            <rdf:li>museum</rdf:li>
            <rdf:li>journalist</rdf:li>
            <rdf:li>system</rdf:li>
            <rdf:li>c</rdf:li>
            <rdf:li>procedures</rdf:li>
            <rdf:li>design</rdf:li>
            <rdf:li>memory</rdf:li>
            <rdf:li>secure</rdf:li>
            <rdf:li>university</rdf:li>
            <rdf:li>higher</rdf:li>
            <rdf:li>plants</rdf:li>
            <rdf:li>flowers</rdf:li>
            <rdf:li>flower</rdf:li>
            <rdf:li>food</rdf:li>
            <rdf:li>garden</rdf:li>
            <rdf:li>gardens</rdf:li>
            <rdf:li>medicine</rdf:li>
            <rdf:li>art</rdf:li>
            <rdf:li>collection</rdf:li>
            <rdf:li>herbaceous</rdf:li>
            <rdf:li>area</rdf:li>
            <rdf:li>rock</rdf:li>
            <rdf:li>running</rdf:li>
         </rdf:Bag>
      </wc:Keyword>
      <wc:Classmark>
         <rdf:Seq>
            <rdf:li>710   Civic and landscape Art</rdf:li>
            <rdf:li>630   Agriculture and Related Technologies</rdf:li>
         </rdf:Seq>
      </wc:Classmark>
      <wc:Word_count>783</wc:Word_count>
      <wc:Classification_date>24-Sep-99 5:37:08 PM</wc:Classification_date>
      <wc:Last_modified>25-Aug-98 10:17:32 AM</wc:Last_modified>
   </rdf:Description>
</rdf:RDF>
```

L.4 Obituary Nevill Mott - http://gordon.cryst.bbk.ac.uk/BCA/obits/nm.html

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://www.scit.wlv.ac.uk/~ex1253/wc/schema">
   <rdf:Description about ="http://gordon.cryst.bbk.ac.uk/BCA/obits/nm.html">
      <wc:Accession_no>0</wc:Accession_no>
      <wc:Title>Obituary Nevill Mott</wc:Title>
      <wc:Abstract>SIR NEVILL MOTT Nevill Francis Mott physicist was born 30 September
         1905 and died 8 August 1996 His work in theoretical solid state physics
         has</wc:Abstract>
      <wc:Keyword>
         <rdf:Bag>
            <rdf:li>professor</rdf:li>
            <rdf:li>education</rdf:li>
            <rdf:li>government</rdf:li>
            <rdf:li>processes</rdf:li>
            <rdf:li>labour</rdf:li>
            <rdf:li>administration</rdf:li>
            <rdf:li>science</rdf:li>
            <rdf:li>sciences</rdf:li>
            <rdf:li>biology</rdf:li>
            <rdf:li>physics</rdf:li>
            <rdf:li>applied</rdf:li>
         </rdf:Bag>
      </wc:Keyword>
      <wc:Classmark>
         <rdf:Seq>
            <rdf:li>330   Economics</rdf:li>
            <rdf:li>530   Physics</rdf:li>
         </rdf:Seq>
      </wc:Classmark>
      <wc:Word_count>298</wc:Word_count>
      <wc:Classification_date>24-Sep-99 5:38:42 PM</wc:Classification_date>
      <wc:Last_modified>09-Jan-98 1:41:41 AM</wc:Last_modified>
   </rdf:Description>
</rdf:RDF>
```

L.5 An Introduction to French Slang - http://orac.sunderland.ac.uk/~os0tmc/mireille/satellite.htm

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://www.scit.wlv.ac.uk/~ex1253/wc/schema">
   <rdf:Description about
      ="http://orac.sunderland.ac.uk/~os0tmc/mireille/satellite.htm">
      <wc:Accession_no>0</wc:Accession_no>
      <wc:Title>An Introduction to French Slang</wc:Title>
      <wc:Abstract>An Introduction to French Slang When time comes for you to cross
         the Channel you might encounter over there a few difficulties regarding the
         vocabulary</wc:Abstract>
      <wc:Keyword>
         <rdf:Bag>
            <rdf:li>university</rdf:li>
            <rdf:li>school</rdf:li>
            <rdf:li>language</rdf:li>
            <rdf:li>french</rdf:li>
         </rdf:Bag>
      </wc:Keyword>
      <wc:Classmark>
         <rdf:Seq>
            <rdf:li>410   Linguistics</rdf:li>
            <rdf:li>440   Romance Languages, French</rdf:li>
         </rdf:Seq>
      </wc:Classmark>
      <wc:Word_count>98</wc:Word_count>
      <wc:Classification_date>24-Sep-99 5:39:39 PM</wc:Classification_date>
      <wc:Last_modified>11-Jun-98 1:55:37 PM</wc:Last_modified>
   </rdf:Description>
</rdf:RDF>
```

L.6 What is Morris Dancing - http://users.zetnet.co.uk/jprice/samm/sammmd.htm

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://www.scit.wlv.ac.uk/~ex1253/wc/schema">
   <rdf:Description about ="http://users.zetnet.co.uk/jprice/samm/sammmd.htm">
      <wc:Accession_no>0</wc:Accession_no>
      <wc:Title>What is Morris Dancing</wc:Title>
      <wc:Abstract>St Albans Morris Men Go to Home Page What is Morris Dancing Do you
         want the 1 minute answer Or have you got a year</wc:Abstract>
      <wc:Keyword>
         <rdf:Bag>
            <rdf:li>community</rdf:li>
            <rdf:li>groups</rdf:li>
            <rdf:li>folk</rdf:li>
            <rdf:li>dancing</rdf:li>
            <rdf:li>dance</rdf:li>
            <rdf:li>play</rdf:li>
         </rdf:Bag>
      </wc:Keyword>
      <wc:Classmark>
         <rdf:Seq>
            <rdf:li>780   Music</rdf:li>
            <rdf:li>790   Recreational and Performing Arts</rdf:li>
         </rdf:Seq>
      </wc:Classmark>
      <wc:Word_count>397</wc:Word_count>
      <wc:Classification_date>24-Sep-99 5:41:25 PM</wc:Classification_date>
      <wc:Last_modified>11-Jul-99 7:17:51 AM</wc:Last_modified>
   </rdf:Description>
</rdf:RDF>
```

APPENDIX M
Source Code for the RDF Metadata Generator.

rdfdocument
The automatic metadata generating version of the classifier uses this extended version of the document object.

```
// This class provides all the methods and instance variables needed to create and
// maintain a document object.  Additional methods and variables are provided for
// generating an RDF description of the document.
package jac;

import java.io.*;
import java.util.*;
import java.net.*;

public class rdfdocument
{

   // Constructor takes an open DataInputStream and an integer representing the
   // accession number as parameters and indexes the file to produce a vector
   // of keywords.
   public rdfdocument (DataInputStream dis, int n) throws IOException
   {

      accession = n;
      docfile = dis;

      doindexing();
      docfile.close();
   }


   // resetKeywords resets a global variables that is used as an index for providing
   // remote access to the keywords vector.
   public void resetKeywords()
   {
      marker = 0;
   }


   // resetClassmarks resets a global variable that is used as an index for providing
   // remote access to the classmarks vector.
   public void resetClassmarks()
   {
      classmarkmarker =0;
   }



   // getAccession returns the accession number
   public int getAccession()
   {
      return accession;
   }


   // getTotal score returns the value of all the keyword scores added together.
   public int getTotalScore()
   {
      return totalScore;
   }


   // getTotal returns the length of the document - the number of keywords in
   // the vector.
   public int getTotal()
   {
      keywords.trimToSize();
      return keywords.size();
   }
```

```
// getWordCount returns the wordCount - representing words on the page not
// in the title or meta tag - used for the RDF description wordcount.
public int getWordCount()
{
   return wordCount;
}



// hasMoreKeywords returns a boolean indicating whether all the keywords
// have been returned
public boolean hasMoreKeywords()
{
   return marker < keywords.size();
}
```

```
// hasMoreClassmarks returns a boolean indicating whether all the classmarks
// have been returned
public boolean hasMoreClassmarks()
{
   return classmarkmarker < classmarks.size();
}



// getNextKeyword returns the next keyword in the vector and increments
// the marker
public keyword getNextKeyword()
{
   if (marker < keywords.size())
      return (keyword)keywords.elementAt(marker++);
   else
      return null;
}


// getNextClassmark returns the next classmark in the vector and increments
// the marker
public String getNextclassmark()
{
   classmark alfie;

   if (classmarkmarker < classmarks.size())
   {
      alfie = (classmark)classmarks.elementAt(classmarkmarker++);
      return (alfie.getClassmark() + "\t" + alfie.getLabel());
   }
   else
      return null;
}


// addClassmark adds a classmark, representing a DDC class that has a significant
// measure of similarity with the document, to the classmarks vector.
public void addClassmark(classmark classMark)
{
   classmarks.addElement(classMark);
}



// getTitle returns the titletext for the RDF description
public String getTitle()
{
     return (titletext);
}


// getAbstract returns the abstracttext for the RDF description
public String getAbstract()
{
     return (abstracttext);
}
```

```
// doindexing reads each line from the file, extracts individual
// words and adds them to the keywords vector along with an appropriate
// score (set according to where it is found) and an integer representing
// its position.
private void doindexing() throws IOException
{
   String line, word, nbspWord, noSCWord, remove = "";
   StringTokenizer words, noBreakSpaceWords, noSemiColonWords;

   line = docfile.readLine();

   // while not end of file
   while (line != null) {

      line = noHTML( line );
      words = new StringTokenizer( line, "<>_+*%\\/@~|.,?:!\"(){}[]-= \t\n\r" );

      while( words.hasMoreTokens() ) {

                word = words.nextToken();

                // test for no break space character entities
                word = processNoBreakSpace( word );

                noBreakSpaceWords = new StringTokenizer( word, " " );

                while( noBreakSpaceWords.hasMoreTokens() ) {

                    nbspWord = noBreakSpaceWords.nextToken();

                    // test for character entities in word
                    if ( ( nbspWord.indexOf( "&" ) > -1 ) && ( nbspWord.indexOf( ";" ) > -
1 ) ) {

                        nbspWord = processCharacterEntities( nbspWord );
                        remove = "";

                    }
                    else if ( nbspWord.indexOf( ";" ) > -1 ) {

                        // set flag remove so all semicolons are removed from nbspWord
                        remove = ";";

                    }
                    else {

                        // word does not need tokenizing
                        remove = "";
                    }


                    // check if word needs retokenizing
                    if ( remove.length() == 0 ) {

                        if ( nbspWord.length() > 0 )
                            addWord(nbspWord);
                    }
                    else {

                        noSemiColonWords = new StringTokenizer( nbspWord, remove );

                        while ( noSemiColonWords.hasMoreTokens() ) {

                            noSCWord = noSemiColonWords.nextToken();
                            if (noSCWord.length() > 0)
                                addWord(noSCWord);
                        }
                    }
                }
            }
        }
        line = docfile.readLine();
    }
    keywords.trimToSize();
}

// noHTML takes a String (line read from the document) as a parameter and
```

```
// returns the same string with the HTML removed.  If a title or heading tag
// is encountered a marker is inserted which is used to increase scores when
// the word is added to the vector.  The contents of any description or keywords
// meta tags are added to the keywords vector directly.
private String noHTML(String s)
{
   int start=0, stop=1;
   String newstring=" ", sub, element=" ", subelement, word1, word2, word3, word4;
   StringTokenizer words;

   s = s + " ";
   while (stop < s.length())
   {
      sub = s.substring(start, stop);
      if (sub.equals("<") || (startHtmlTag && !endHtmlTag))
      {
         if (sub.equals("<"))
      {
         startHtmlTag = true;
         endHtmlTag = false;
      }
      while (stop < s.length() && !sub.equals(">"))
      {
         element = element + sub;
         stop++;
         start++;
         sub = s.substring(start,stop);
       }
       if (sub.equals(">"))
       {
          startHtmlTag = false;
          endHtmlTag = true;
       }
       start++;
       stop++;
       subelement = element.substring(2, element.length());
       if (subelement.equalsIgnoreCase("TITLE"))
         newstring = newstring + " jacmarker1 ";
       else if (subelement.equalsIgnoreCase("/TITLE"))
         newstring = newstring + " jacmarkerend ";
       else if (subelement.equalsIgnoreCase("H1"))
         newstring = newstring + " jacmarkerh1 ";
       else if (subelement.equalsIgnoreCase("/H1"))
         newstring = newstring + " jacmarkerh1end ";
       else if (subelement.equalsIgnoreCase("H2"))
         newstring = newstring + " jacmarkerh2 ";
       else if (subelement.equalsIgnoreCase("/H2"))
         newstring = newstring + " jacmarker2end ";
       else newstring = newstring + " ";
       element = " ";
       words = new StringTokenizer(subelement, "_+*%\\/@~|.,?:!\\'\"(){}[]=
\t\n\r");
       if (words.countTokens() > 4)
       {
          word1 = words.nextToken();
          word2 = words.nextToken();
          word3 = words.nextToken();
          word4 = words.nextToken();
          if ((word1.equalsIgnoreCase("meta")) &&
(word2.equalsIgnoreCase("name")) && (word3.equalsIgnoreCase("keywords")) &&
(word4.equalsIgnoreCase("content")))
             meta = true;
          else if ((word1.equalsIgnoreCase("meta")) &&
(word2.equalsIgnoreCase("name")) && (word3.equalsIgnoreCase("description")) &&
(word4.equalsIgnoreCase("content")))
       {
          meta = true;
          description = true;
       }
       if (meta)
       {
          score += 9;
          while (words.hasMoreTokens())
             addWord(words.nextToken());
          score -= 9;
          if (endHtmlTag)
          {
```

```
                    meta=false;
                    description=false;
                }
            }
        }
        }
        else
        {
            newstring = newstring + sub;
            start++;
            stop++;
        }
    }
    return newstring;
}


private String processNoBreakSpace( String word ) {

    String temp, tempEnd;
    int i, index;


    // loop through all no break space codes
    for ( i =0; i < nbspCodes.length; i++ ) {

        // test if word contains a no break space code
        while (( index = word.indexOf( nbspCodes[i] )) > -1 ) {

            if ( index == 0 ) {

                temp = word.substring( nbspCodes[i].length(), word.length() );
                word = temp;
            }
            else if ( index == ( word.length() - nbspCodes[i].length() )) {

                temp = word.substring( 0, index );
                word = temp;
            }
            else {
                    temp = word.substring( 0, index );
                    tempEnd = word.substring( index + nbspCodes[i].length(),
word.length() );
                    word = temp + " " + tempEnd;
                }
            }
        }


        return word;
    }


    private String processCharacterEntities( String word ) {

        boolean stop = false;
        int ampersand = 0, secondAmp = 0, semicolon = 0, secondSemi = 0;
        String entity, temp;


        // check for character entities at the start of the word
        while ( word.startsWith("&") && !stop ) {

        semicolon = word.indexOf( ';' );
        secondAmp = word.indexOf( '&', 1 );

        // test if ampersand is part of a character entity
        if (( secondAmp > -1 ) && ( secondAmp < semicolon )) {

            stop = true;

        }
        else {
```

```
// take a copy of the entity
entity = word.substring( 0, semicolon + 1 );

if ( !validCharacterEntity( entity )) {

    // remove entity
    // test for the end of the word
    if ( semicolon == word.length() - 1 ) {
        temp = "";
    }
    else {
        temp = word.substring( semicolon + 1, word.length() );
    }
    word = temp;

}
else {
    stop = true;
}
        }
    }

    // check if the end of the word has been reached
    if ( semicolon != word.length() - 1 ) {

        stop = false;

        // check for character entites at the end of the word
        while ( word.endsWith(";") && !stop ) {

            ampersand = word.lastIndexOf( '&' );
            secondSemi = word.lastIndexOf( ';', word.length() - 2 );


            // test for semicolons that are not part of the entity
            if ( (ampersand < 0) || (( secondSemi > -1 ) && ( secondSemi >
ampersand )) ) {

                temp = word.substring( 0, word.length() - 1 );
                word = temp;

            }
            else {

                // take a copy of the entity
                entity = word.substring( ampersand, word.length() );

                if ( !validCharacterEntity( entity )) {

                    // remove entity
                    temp = word.substring( 0, ampersand );
                    word = temp;

                }
                else {
                    stop = true;
                }
            }
        }
    }
    return word;
}

private boolean validCharacterEntity( String entity ) {

String strNum;
int num, i;

// test for numerical character entity
if ( entity.charAt( 1 ) == '#' ) {

    strNum = entity.substring( 2, entity.length() - 1 );
    num = Integer.parseInt( strNum );

    if ( ( num == 38 ) ||
         ( num >= 48 && num <= 57 ) ||
         ( num >= 65 && num <= 90 ) ||
```

```
                ( num >= 97 && num <= 122 ) ||
                ( num >= 192 && num <= 214 ) ||
                ( num >= 216 && num <= 246 ) ||
                ( num >= 248 && num <= 255 ) ) {

            return true;
        }
        else {

            return false;
        }
    }
    else {

        for ( i = 0; i < alphaCharEntities.length; i++ ) {

            if ( entity.compareTo( alphaCharEntities[i] ) == 0 ) {
                return true;
            }
        }

        return false;
    }
}


// addWord adds the word passed as a parameter to the vector along with integers
// representing its score and position.  Scores are increased if the word is found
// within a title, heading or meta tag.
// If the word is found in the title, meta description or first 25 words of
// the page (in the absence of a meta description) the word is appended to
// titletext or abstracttext respectively for use in the RDF description.
private void addWord(String word)
{
   if (word.equals("jacmarker1"))
   {
      if (!past_title)
      {
         score += 9;
         past_title = true;
         title = true;
   }
}
else if (word.equals("jacmarkerend"))
{
   if (title)
   {
      score -= 9;
      title = false;
   }
}
else if (word.equals("jacmarkerh1"))
{
   if (!heading)
   {
      score += 9;
      heading = true;
   }
}
else if (word.equals("jacmarkerh1end"))
{
   if (heading)
   {
      score -= 9;
      heading = false;
   }
}
else if (word.equals("jacmarkerh2"))
{
   if (!heading)
   {
      score += 4;
      heading = true;
   }
```

```
}
else if (word.equals("jacmarker2end"))
{
   if (heading)
   {
      score -= 4;
      heading = false;
   }
}
else
{
   wordnumber++;
   keywords.addElement(new keyword(word, score, wordnumber));
   totalScore += score;
   if (!title && !meta)
      wordCount++;
   if (title)
   {
      if (titletext == null) titletext = word;
      else titletext = titletext + " " + word;
   }
   else
   {
      if (!meta)
      {
         if ((!abstractdone) && (abstractcount==0))
         {
            abstracttext = word;
            abstractcount++;
         }
         else if ((!abstractdone) && (abstractcount < 25))
         {
            abstracttext = abstracttext + " " + word;
            abstractcount++;
         }
      }
      else
      {
         if (description)
         {
            if (abstractcount == 0)
            {
               abstractdone = true;
               abstracttext = word;
               abstractcount++;
            }
            else
            {
               abstracttext = abstracttext + " " + word;
               abstractcount++;
            }
         }
      }
   }
}

   private DataInputStream docfile;
   private Vector keywords = new Vector(20,50);
   private Vector classmarks = new Vector(5,5);
   private int accession, totalScore=0, classmarkmarker=0, marker=0, abstractcount=0,
wordCount=0;
   private int wordnumber=0, score=1;
   private String titletext=null, abstracttext=null;
   private boolean past_title=false, title = false, heading=false, meta=false,
description=false, abstractdone=false;
   private static boolean startHtmlTag = false, endHtmlTag = false;
   private String[] nbspCodes = { " ", " " };
   private String[] alphaCharEntities = { "&Agrave;", "&Aacute;", "&Acirc;",
"&Atilde;", "&Auml;", "&Aring;", "&AElig;", "&Ccedil;", "&Egrave;", "&Eacute;",
"&Ecirc;", "&Euml;", "&Igrave;", "&Iacute;", "&Icirc;", "&Iuml;", "&ETH;", "&Ntilde;",
"&Ograve;", "&Oacute;", "&Ocirc;", "&Otilde;", "&Ouml;", "&Oslash;", "&Ugrave;",
"&Uacute;", "&Ucirc;", "&Uuml;", "&Yacute;", "&THORN;", "&szlig;", "&agrave;",
"&aacute;", "&acirc;", "&atilde;", "&auml;", "&aring;", "&aelig;", "&ccedil;",
"&egrave;", "&eacute;", "&ecirc;", "&euml;", "&igrave;", "&iacute;", "&icirc;",
"&iuml;", "&eth;", "&ntilde;", "&ograve;", "&oacute;", "&ocirc;", "&otilde;",
```

```
"&ouml;", "&oslash;", "&ugrave;", "&uacute;", "&ucirc;", "&uuml;", "&yacute;",
"&thorn;", "&yuml;" };

}
```

rdfclassify

The automatic metadata generating version of the classifier uses this extended version of the classify object.

```
// This class takes a document object (as a parameter on the constructor) and proceeds
// to classify it by comparing it with each branch of the DDC hierarchy.
// This version maintains a list of significant DDC classes so that it has access to
// significant keywords for RDF metadata generation.  Only the classmarks of the two
// highest scoring classes are assigned to the document.
package jac;

import java.io.*;
import java.util.*;
import jac.deweydecimal.*;
import jac.deweydecimal.generalities.*;
import jac.deweydecimal.philosophy.*;
import jac.deweydecimal.religion.*;
import jac.deweydecimal.socialsciences.*;
import jac.deweydecimal.language.*;
import jac.deweydecimal.naturalsciences.*;
import jac.deweydecimal.technology.*;
import jac.deweydecimal.arts.*;
import jac.deweydecimal.literature.*;

// Constructor takes a document as a parameter and calls the proceed method for
// each branch of the hierarchy.
public class rdfclassify
{

   public rdfclassify(rdfdocument d)
   {
      Vector wordvector = new Vector (20,10);
      doc = d;
      proceed(new generalitiesclass(), wordvector);
      proceed(new philosophyclass(), wordvector);
      proceed(new religionclass(), wordvector);
      proceed(new socialsciencesclass(), wordvector);
      proceed(new languageclass(), wordvector);
      proceed(new naturalsciencesclass(), wordvector);
      proceed(new technologyclass(), wordvector);
      proceed(new artsclass(), wordvector);
      proceed(new literatureclass(), wordvector);
/*      proceed(new geoghistory(), wordvector); not implemented */
      assignClassmarks();

   }


   // proceed takes a dewey object as a paramter and scores it against
   // (compares it with) the document.  If the score is significant, the
   // proceed method is then called recursively for each of any subclasses.
   private void proceed(dewey ddc, Vector sigwords)
   {
      classmark cm;
      int totalscore;
      significantclass sigclass;

      totalscore = score(ddc);
      if (significant(totalscore, ddc.getTotal(), doc.getTotal()))
      {
         sigwords = copysigwords (sigwords, scoredwords);
         if (!ddc.hasMoreSubclasses())
         {
            cm = ddc.getClassmark();
            cm.setScore(totalscore);
            sigclass = new significantclass(sigwords, cm);
            sigclasses.addElement(sigclass);
         }
         else
         {
```

```
            while (ddc.hasMoreSubclasses())
                proceed(ddc.getNextSubclass(), sigwords);
         }
      }
      scoredwords = removewords(scoredwords);
   }


   // significant takes the total score associated with a document/class
   // comparison, the length of the class and the length of the document
   // and calculates the Dice Coefficient.
   private boolean significant(int totalscore, int deweylength, int doclength)
   {
      float totalLength = deweylength + doclength;

      if ((2 * (totalscore / totalLength)) > 0.5)
         return true;
      else
         return false;
   }


   // score takes a dewey object as a parameter and compares each word in the
   // dewey keyword vector with each word in the document keyword vector resulting
   // in a total score.
   private int score(dewey ddc)
   {
      int thescore=0,doccount=0, deweycount=0;
      keyword docword, deweyword;

      while (ddc.hasMoreKeywords())
      {
         deweyword = ddc.getNextKeyword();
         while (doc.hasMoreKeywords())
         {
            docword = doc.getNextKeyword();
            if (deweyword.is_equal(docword))
            {
               thescore = thescore + deweyword.getScore() + docword.getScore();
               scoredwords.addElement(new String(deweyword.getKeyword()));
            }
         }
         doc.resetKeywords();
      }
      scoredwords.trimToSize();
      return thescore;
   }


   // removewords removes words from the words vector (used in obtaining
   // significant keywords for the RDF description).
   private Vector removewords(Vector words)
   {
      while (words.size() > 0)
        words.removeElementAt(0);
      return words;
   }


   // copysigwords joins two vectors of significant keywords and returns the
   // resulting vector
   private Vector copysigwords(Vector significant, Vector toadd)
   {
      int marker = 0;

      while (marker < toadd.size())
         significant.addElement(new String((String)toadd.elementAt(marker++)));
      significant.trimToSize();
      return significant;
   }


   // assignClassmarks works out which of a vector of classmarks is the highest
   // scoring and assigns the two highest to the document
   private void assignClassmarks()
```

```
{
   int marker=0;
   significantclass one, two, current;
   Vector words;

   if (marker < sigclasses.size())
   {
      one = (significantclass)sigclasses.elementAt(marker++);
      two = one;
      while (marker < sigclasses.size())
      {
         current = (significantclass)sigclasses.elementAt(marker++);
         if(current.getScore() >= one.getScore())
         {
            two = one;
            one = current;
         }
         else if (current.getScore() >= two.getScore())
            two = current;
      }
      doc.addClassmark(two.getclassmark());
      doc.addClassmark(one.getclassmark());
      addSignificantwords(one.getkeywords());
      addSignificantwords(two.getkeywords());
   }
}


// addSignificantwords adds the contents of the vector passed as a
// parameter to the sigkeywords vector.
private void addSignificantwords(Vector words)
{
   int marker = 0;
   String currentword;

   while (marker < words.size())
   {
      currentword = (String)words.elementAt(marker++);
      if (notalreadysig(currentword))
         sigkeywords.addElement(new String(currentword));
   }
   sigkeywords.trimToSize();
}


// notalreadysig returns a boolean indicating whether the word
// passed as a parameter is already in the sigkeywords vector.
private boolean notalreadysig(String word)
{
   int marker = 0;

   while (marker < sigkeywords.size())
   {
      if (word.equalsIgnoreCase((String)sigkeywords.elementAt(marker++)))
         return false;
   }
   return true;
}


// hasMoreSigkeywords returns a boolean indicating whther all the significant
// keywords have been returned from the sigkeywords vector
public boolean hasMoreSigkeywords()
{
   return sigkeywordmarker < sigkeywords.size();
}


// getNextSigKeyword returns the next significant word from the vector - used
// for generating keywords in the RDF description.
public String getNextSigKeyword()
{
   if (sigkeywordmarker < sigkeywords.size())
      return (String)sigkeywords.elementAt(sigkeywordmarker++);
   else
      return null;
}
```

```
// resetSigKeywords returns the sigkeywordmarker to 0 so the significant keywords
// can be retrieved again if required.
public void resetSigKeywords()
{
   sigkeywordmarker = 0;
}



private Vector sigkeywords = new Vector(20,10);
private Vector sigclasses = new Vector (20,10);
private Vector scoredwords = new Vector (20,10);
private int sigkeywordmarker = 0;
private rdfdocument doc;
}
```

rdface

The automatic metadata generating version of the classifier uses this extended version of the ACE object:

```
// This class co-ordinates the classification process by opening the document,
// generating a document object and passing that document object as a parameter to
// an instance of the classify object.  It then uses methods within the document and
// classify object to generate an RDF description of the document.

import jac.*;
import java.io.*;
import java.util.*;
import java.net.*;


public class rdface
{
   public static void main (String[] args)
   {
      DataInputStream docfile;
      URL url;
      HttpURLConnection uc;
      long lastModLong = 0;


      if (args.length == 0)
         System.out.println("Usage: java ace <filename>\nUsage: java ace -url <URL>");
      else
      {
         if (args[0].equals("-url"))
         {
            if (args.length < 2)
            {
               System.out.println("Usage: java ace <filename>\nUsage: java ace -url
<URL>");
               System.exit(1);
            }
            else
            {

               try
               {
                  url = new URL(args[1]);
                  remote=true;
                  uc = (HttpURLConnection) url.openConnection();
                  lastModLong = uc.getLastModified();
                  lastMod = new Date(lastModLong);
                  docfile = new DataInputStream(url.openStream());
                  Doc = new rdfdocument(docfile,0);
                  classification = new rdfclassify(Doc);
                  outputrdf(args);
                  docfile.close();

               }
               catch (MalformedURLException mu)
               {
```

```
            System.out.println ("Sorry cannot find URL: " + args[1]);
            System.exit(1);
         }
         catch (IOException e)
         {
            System.out.println ("Sorry cannot connect to URL: " + args[1]);
            System.exit(1);
         }
      }

   }
   else
   {
      try
      {
         docfile = new DataInputStream(new FileInputStream(args[0]));
         Doc = new rdfdocument(docfile,0);
         classification = new rdfclassify(Doc);
         lastMod = new Date(0);
         outputrdf(args);
         docfile.close();
      }
      catch (IOException e)
      {
          System.out.println ("Error reading file " + args[0]);
          System.exit(1);
      }
    }
  }
  }

  // outputrdf extracts all the necessary information from Doc and classification to
  // generate an RDF description of the document.
  private static void outputrdf(String[] args)
  {

  Date today = new Date();
  Date epoch = new Date(0);
  System.out.print("<?xml version=\"1.0\"?>\n<rdf:RDF\n
xmlns:rdf=\"http://www.w3.org/TR/WD-rdf-syntax#\"\n
xmlns:wc=\"http://scit.wlv.ac.uk/~ex1253/wc/schema/\">\n");
  System.out.print("\t<rdf:Description about =\"");
  if (remote) System.out.print(args[1]);
  else System.out.print(args[0]);
  System.out.print("\">\n\t\t<wc:Accession_no>" + Doc.getAccession() +
  "</wc:Accession_no>\n\t\t<wc:Title>" + Doc.getTitle() +
"</wc:Title>\n\t\t<wc:Abstract>" + Doc.getAbstract() + "</wc:Abstract>\n");
  System.out.print("\t\t<wc:Keyword>");
  if (classification.hasMoreSigkeywords())
  {
     System.out.print("\n\t\t\t<rdf:Bag>\n");
     while (classification.hasMoreSigkeywords())
        System.out.print("\t\t\t\t<rdf:li>" + classification.getNextSigKeyword() +
"</rdf:li>\n");
     System.out.print("\t\t\t</rdf:Bag>\n\t\t</wc:Keyword>\n");
  }
  else
     System.out.print("Keywords not identified</wc:Keyword>\n");
  System.out.print("\t\t<wc:Classmark>");
  if (Doc.hasMoreClassmarks())
  {
     System.out.print("\n\t\t\t<rdf:Seq>\n");
     while (Doc.hasMoreClassmarks())
        System.out.print("\t\t\t\t<rdf:li>" + Doc.getNextclassmark() +
"</rdf:li>\n");
     System.out.print("\t\t\t</rdf:Seq>\n\t\t</wc:Classmark>\n");
  }
  else
     System.out.print("Classification not identified</wc:Classmark>\n");
  System.out.print("\t\t<wc:Word_count>" +
  Doc.getWordCount() + "</wc:Word_count>\n");
  System.out.print("\t\t<wc:Classification_date>" +
  today.toLocaleString() + "</wc:Classification_date>\n");
  System.out.print("\t\t<wc:Last_modified>");
  if (lastMod.equals(epoch))
     System.out.print("Not known");
  else
```

```
      System.out.print(lastMod.toLocaleString());
   System.out.print("</wc:Last_modified>\n\t</rdf:Description>\n</rdf:RDF>\n");
   }

   private static boolean remote=false;
   private static rdfdocument Doc = null;
   private static rdfclassify classification;
   private static Date lastMod;

}
```

APPENDIX N
Source code of the Metadata Generating Servlet.

There follows the source code of the Java Servlet used to implement the on-line version of the Automatic Metadata Generator shown in figure 51 (section 5.4). This class replaces rdface. It enables the option to view just the DDC classmarks, RDF using Dublin Core or RDF using Wolverhampton Core.

```
import java.io.*;
import java.util.*;
import java.net.*;
import javax.servlet.*;
import javax.servlet.http.*;
import jac.*;

public class metadatagenerator extends HttpServlet
{
    public void doPost(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException
    {
        DataInputStream docfile;
        HttpURLConnection uc;
        long lastModLong = 0;
        URL location;
        String urlString = null;
        res.setContentType("text/html");
        PrintWriter toClient = res.getWriter();

        toClient.println("<HTML>");
        toClient.println("<HEAD>");
        toClient.println("<TITLE>Automatically Generated Metadata</TITLE>");
        toClient.println("</HEAD>");
        toClient.println("<BODY BGCOLOR=\"#FFFFFF\">");
        toClient.println("<CENTER>");
        toClient.println("<H1>Automatically Generated Metadata</H1>");
        toClient.println("</CENTER>");
        toClient.println("<HR>");

        try
        {
                urlString=req.getParameterValues("location")[0];
                location = new URL(urlString);
                uc = (HttpURLConnection) location.openConnection();
                lastModLong = uc.getLastModified();
                lastMod = new Date(lastModLong);
                docfile = new DataInputStream(location.openStream());
                Doc = new rdfdocument(docfile,0);
                classification = new rdfclassify(Doc);

                String metadatatype = req.getParameterValues("option")[0];
                if (metadatatype.equals("ddc"))
                        doDDCoutput(toClient);
                else
                        doWCoutput(toClient, urlString);
                docfile.close();
        }
        catch (MalformedURLException mu)
        {
           toClient.println("Sorry cannot find URL: " + urlString);
        }
        catch (IOException e)
        {
           toClient.println("Sorry cannot connect to URL: " + urlString);
        }

        toClient.println("<HR>");
        toClient.println("<P>Please <A HREF=\"mailto:ex1253@wlv.ac.uk\">email your
comments</A> to Charlotte Jenkins</P>");
        toClient.println("</BODY>");
        toClient.println("</HTML>");
        toClient.close();
    }
```

```
private void doDDCoutput(PrintWriter out)
{
    out.println("<H2>DDC Classmarks only</H2>");
    out.println("<P>");
    if (Doc.hasMoreClassmarks())
    {
          while (Doc.hasMoreClassmarks())
                out.println(Doc.getNextclassmark()\n);
    }
    else
          out.println("Classification not identified");
    out.println("</P>");
}


private void doWCoutput(PrintWriter out, String url)
{
    out.println("<H2>RDF using Wolverhampton Core</H2>");
    out.println("<P>");
        out.println("<pre>");
        Date today = new Date();
        Date epoch = new Date(0);
        out.print("&lt;?xml version=\"1.0\"?&gt;\n&lt;rdf:RDF\n
xmlns:rdf=\"http://www.w3.org/TR/WD-rdf-syntax#\"\n
xmlns:wc=\"http://scit.wlv.ac.uk/~ex1253/wc/schema/\"&gt;\n");
        out.print("\t&lt;rdf:Description about =\"" + url + "\"&gt;\n");
        out.print("\t\t&lt;wc:Accession_no&gt;" + Doc.getAccession() +
        "&lt;/wc:Accession_no&gt;\n\t\t&lt;wc:Title&gt;" + Doc.getTitle() +
"&lt;/wc:Title&gt;\n\t\t&lt;wc:Abstract&gt;" + Doc.getAbstract() +
"&lt;/wc:Abstract&gt;\n");
        out.print("\t\t&lt;wc:Keyword&gt;");
        if (classification.hasMoreSigkeywords())
        {
           out.print("\n\t\t\t&lt;rdf:Bag&gt;\n");
           while (classification.hasMoreSigkeywords())
              out.print("\t\t\t\t&lt;rdf:li&gt;" + classification.getNextSigKeyword()
+ "&lt;/rdf:li&gt;\n");
           out.print("\t\t\t&lt;/rdf:Bag&gt;\n\t\t&lt;/wc:Keyword&gt;\n");
        }
        else
           out.print("Keywords not identified&lt;/wc:Keyword&gt;\n");
        out.print("\t\t&lt;wc:Classmark&gt;");
        if (Doc.hasMoreClassmarks())
        {
           out.print("\n\t\t\t&lt;rdf:Seq&gt;\n");
           while (Doc.hasMoreClassmarks())
              out.print("\t\t\t\t&lt;rdf:li&gt;" + Doc.getNextclassmark() +
"&lt;/rdf:li&gt;\n");
           out.print("\t\t\t&lt;/rdf:Seq&gt;\n\t\t&lt;/wc:Classmark&gt;\n");
        }
        else
           out.print("Classification not identified&lt;/wc:Classmark&gt;\n");
        out.print("\t\t&lt;wc:Word_count&gt;" +
        Doc.getWordCount() + "&lt;/wc:Word_count&gt;\n");
        out.print("\t\t&lt;wc:Classification_date&gt;" +
        today.toLocaleString() + "&lt;/wc:Classification_date&gt;\n");
        out.print("\t\t&lt;wc:Last_modified&gt;");
        if (lastMod.equals(epoch))
           out.print("Not known");
        else
           out.print(lastMod.toLocaleString());

out.print("&lt;/wc:Last_modified&gt;\n\t&lt;/rdf:Description&gt;\n&lt;/rdf:RDF&gt;\n")
;
        out.println("</pre>");
        out.println("</P>");
    }

   private static rdfdocument Doc = null;
   private static rdfclassify classification;
   private static Date lastMod;


}
```

APPENDIX O
Publications associated with the thesis.

This appendix comprises the following papers that were published as a result of the work described in the thesis:

- Jenkins, C., Jackson, M., Burden, P., "Automatic Generation of RDF Metadata". Published in the proceedings of ACM DL'99 Workshop on organising Web Space, Berkeley, California, U.S.A., August 14th. 1999
- Jenkins, C., Jackson, M., Burden, P. and Wallis, J. "Automatic RDF metadata generation for resource discovery". Published in the proceedings of the 8th International World Wide Web Conference, Toronto, 11-14 May 1999 pp 227-242 ISBN: 0-444-50264-5 also in Computer Networks Vol. 31 no. 15 pp 11-16, Elsevier, May 1999.
- (Abstract) Jenkins, C., Jackson, M., Burden, P., and Wallis, J., "The Wolverhampton Web Library (WWLib) and Automatic Classification". Presented at the 1st International Workshop on Libraries and WWW, Brisbane, Queensland, Australia 14th April 1998
- Jenkins, C., Jackson, M., Burden, P., and Wallis, J., "Automatic Classification of Web Resources using Java and Dewey Decimal Classifications". Published in the proceedings of the 7th International World Wide Web Conference, Brisbane, Queensland, Australia 14 - 18 April 1998 also in Computer Networks and ISDN Systems, Vol. 30, Pages: 646-648, ISSN:0169-7552, Elsevier, 1998.
- Jenkins, C., Jackson, M., Burden, P., and Wallis, J., "Searching the World Wide Web: An Evaluation of Available Tools and Methodologies". Published in Information & Software Technology, Vol 39, No14-15, Elsevier, 1998.

Automatic Generation of RDF Metadata

Charlotte Jenkins
School of Computing & IT
University of Wolverhampton
35/49 Lichfield St. Wolverhampton
WV1 1EL, UK
+44 1223 290707
ex1253@wlv.ac.uk

Mike Jackson
School of Computing & IT
University of Wolverhampton
35/49 Lichfield St. Wolverhampton
WV1 1EL, UK
+44 1902 321429
m.s.jackson@wlv.ac.uk

Peter Burden
School of Computing & IT
University of Wolverhampton
35/49 Lichfield St. Wolverhampton
WV1 1EL, UK
+44 1902 321468
jphb@scit.wlv.ac.uk

ABSTRACT

The Resource Description Framework (RDF[9]) has been developed to fulfil the need for a mechanism for resource description within the Web's architecture. With over 320 million[10] individually accessible objects on the Web, the ability to describe each one so that it can be conceptualized without being accessed and analyzed is increasingly important. This paper describes how an automatic classifier[8], that classifies Web pages according to Dewey Decimal Classification[6], can be used to automatically extract various metadata elements in addition to the classification classmarks. This metadata is then presented in RDF format. The classifier is written in Java and has been developed to form part of a distributed automated search engine. The use of automatic classification is intended to combine the well organized, intuitive to use, accurate features of classified directories with the comprehensive coverage and speed of automated search engines. An appropriate metadata element set for use within an automated search engine is defined which is an interoperable subset of the Dublin Core[14] elements. An RDF data model[9] and schema[1] are defined along with examples of automatically generated RDF for a range of resources. Automatically generated metadata of this kind has considerable potential for describing shared resources between subject gateways and could also encourage information sharing between automated search engines. Resource descriptions are also important for content rating and authentication and as such the automatic generation of RDF metadata is arguably an essential prerequisite for a comprehensive 'Web of Trust'.

Keywords

RDF, Metadata, Automatic Classification, Resource Discovery

1. INTRODUCTION

Historically, tools for information resource discovery on the Web have fallen into two categories; manually maintained classified directories comprising a classified hierarchy of manually described resources and automated search engines comprising a huge index of automatically analyzed resources. The former tend to benefit from very accurate, well organized, high quality resource descriptions but suffer from poor Web coverage and an incapacity to cope with the transient nature of the Web (dead links). The latter tend to provide much more comprehensive and up to date Web coverage due to the constant activity of their automated components, but suffer greatly from very poor quality resource descriptions allowing no notion of context. To the end user this means that the classified directories provide good quality but incomplete and often out of date results and the automated search engines usually provide a puzzling and overwhelming set of less than accurate results which are also incomplete, sometimes out of date and often repetitive.

One method of improving resource description within automated search engines is to encourage authors to include embedded metadata, in the form of the HTML meta tag, in their documents. Although this method should improve matters, there are two main reasons why it can never be trusted:

1. Inconsistency - meta tags are not compulsory, therefore not every page has them and very old pages that predate these concerns are not likely to have them in any case.
2. Inaccuracy - if meta tags are there, they are not necessarily accurate.

Most automated search engines tend to ignore meta tags because they can be used as a vehicle for spamming - hiding inappropriate terms to increase the search engine popularity (or ranking) of a particular resource. Also, some authoring tools automatically generate very poor quality and largely irrelevant meta information.

The World Wide Web Consortium (W3C[15]) have introduced the Resource Description Framework (RDF[9]) in an attempt to build a mechanism for resource description into the Web's architecture. RDF will be important for resource discovery but also for content rating (PICS[11]) authentication and intellectual property rights. W3C have a notion of a *Web of Trust* where individually accessible objects on the Web will be well defined and digitally signed using RDF. RDF expressed in XML (the eXtensible Mark-up Language[3]) enables extensible yet interoperable metadata element sets to be defined and interpreted using the XML namespace mechanism and RDF schemas. RDF descriptions can be used to describe individual resources or groups of resources, they can be embedded or linked. The problem with RDF though, is that like the HTML meta tag, its use is optional. Although future authoring tools are likely to encourage the inclusion of RDF descriptions, the current situation, where some pages are described and others are not, is likely to continue for some time. Older pages are unlikely to have

associated RDF descriptions and RDF could just as easily be used to associate an inappropriate description. Digitally signed RDF is intended to enforce the *Web of Trust* so that a user need not accept information unless it is signed with a signature they know they can trust. However, the *Web of Trust* really needs to be comprehensive if it is to realistically protect users, particularly children, from illicit material and dangerous information. Comprehensive, accurate RDF descriptions are required to assist tools for information resource discovery and to properly facilitate the *Web of Trust*.

The following sections describe how an automatic classifier, that classifies Web pages according to Dewey Decimal Classification (DDC), can be used to a) build a notion of context into an automated search engine and b) generate an unbiased RDF description of any HTML document.

2. AUTOMATIC CLASSIFICATION

Most of the major search engines have turned to traditional Information Retrieval (IR) research for ideas on how to improve IR on the Web. Few, however, have experimented with the research on automatic classification that Good[7], Fairthorne[5] and Salton[12] described as long ago as the early 1960's. If the process of classification could be automated then automated search engines could share some of the advantages of classified directories. Documents sharing the same subject matter can be clustered and retrieved together more easily. This would reduce the likelihood of inundating the user with largely irrelevant results. Considering the content of a document in context vastly improves both resource description and resource discovery.

The automatic classifier described in this paper was developed to form one component of a distributed, automated search engine - the Wolverhampton Web Library (WWLib).

2.1 The Wolverhampton Web Library (WWLib)

The original WWLib[2] can best be described as a manually maintained classified directory that arranged UK Web pages according to DDC. Although successful in its day, the original WWLib highlighted the need for a much higher degree of automation. A new fully automated WWLib is currently being developed. Figure 1 shows an overview of the WWLib-TNG (The Next Generation) architecture.

There are essentially six automated components:

1. A Spider that automatically retrieves documents from the Web;
2. An Archiver that receives Web pages from the Spider, stores a local copy, assigns to it a unique accession number and generates a new metadata file. It also distributes local copies to the Extractor, Classifier and Builder and adds subsequent metadata generated by the Classifier and the Builder to the assigned metadata file;
3. An Extractor that analyses pages provided by the Archiver for embedded hyperlinks to other documents. If found, URLs are passed to the Archiver where they are evaluated to check that they are pointing to locations in the UK, before being passed to the Spider;
4. A Classifier (described in the next subsection) that analyses pages provided by the Archiver and generates DDC classmarks;
5. A Builder that analyses pages provided by the Archiver and outputs metadata which is stored by the Archiver in the document's metadata file and is also used to build the index database that will be used to quickly associate keywords with document accession numbers;
6. A Searcher that accepts query strings from the user and uses them to interrogate the index database built by the Builder. It then uses the resulting accession numbers to retrieve the appropriate metadata and local document copies and then uses all this information to generate detailed results, ranked according to relevance to the original query.

The library metaphor has been pursued in the belief that library science, that has been responsible for organizing vast amounts of information for decades, has a lot to offer the comparatively chaotic task of IR on the Web. Classification and metadata, both ideas originating from library science, appear to be the way forward.

2.2 The Automatic Classifier

The automatic classifier classifies HTML pages according to

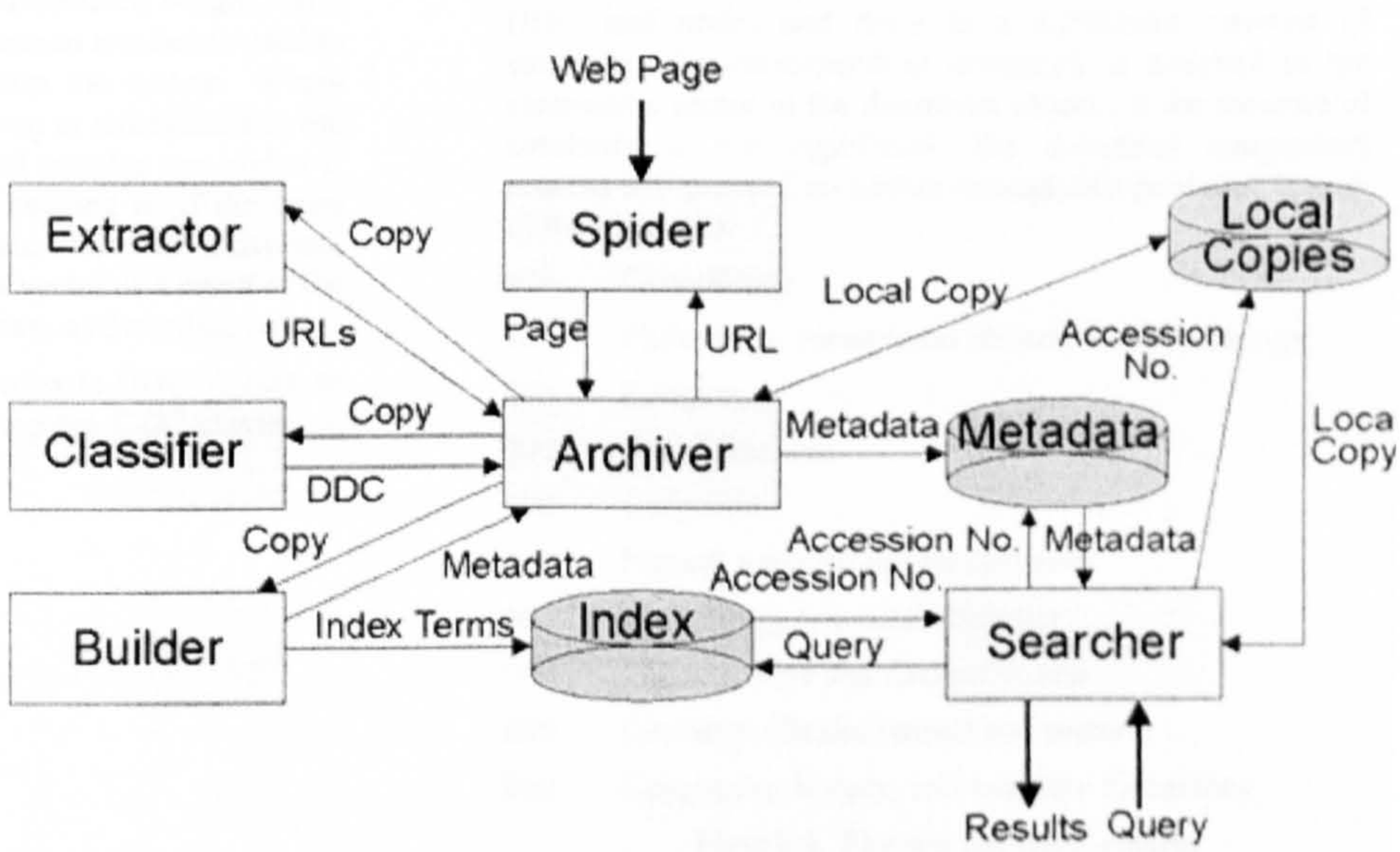


Figure 1. Overview of the WWLib-TNG architecture

DDC. DDC was considered appropriate because it is familiar to anyone (from the UK) accustomed to using a library, it provides universal coverage, has multilingual scope and its hierarchical nature will enable the users of a search engine to refine their search from rough classifications to increasingly more accurate

ones. This is another clear advantage of classified tools for resource discovery, they enable users to browse a classification hierarchy refining their search as they go, making it unnecessary to form a complex Boolean query string to find exactly what they want.

The classifier is an object oriented system written in Java. It works by taking a URL or path to a given local file, parsing the associated HTML and generating appropriate DDC classmarks. There are two main processes involved:

1. Firstly, the HTML document is parsed to produce a document object, as shown in figure 2.

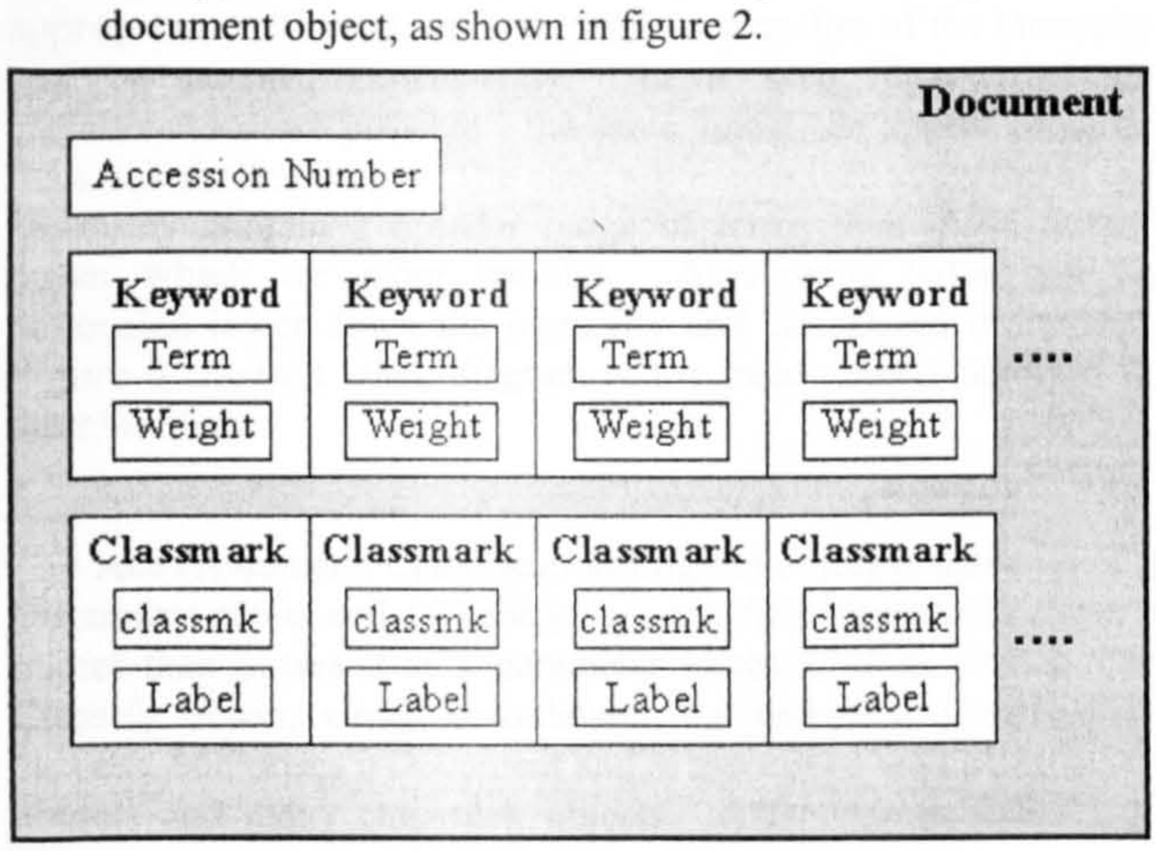


Figure2. A document object

A document object comprises a unique accession number, a series of keyword objects - each one representing a term found within the document with an associated weight and a series of classmark objects. The accession number is used to uniquely identify the document within the system. Every individual word found in the document is represented in the vector of keyword objects. Keyword weights are higher if the word was found in the title, a heading or if the word appears several times. Appropriate, relevant classmark objects are assigned to the classmarks vector as a result of the second stage of the classification process as described below.

2. The document object is then compared with DDC objects, as shown in figure 3, representing the top ten DDC classes, as shown in figure 4.

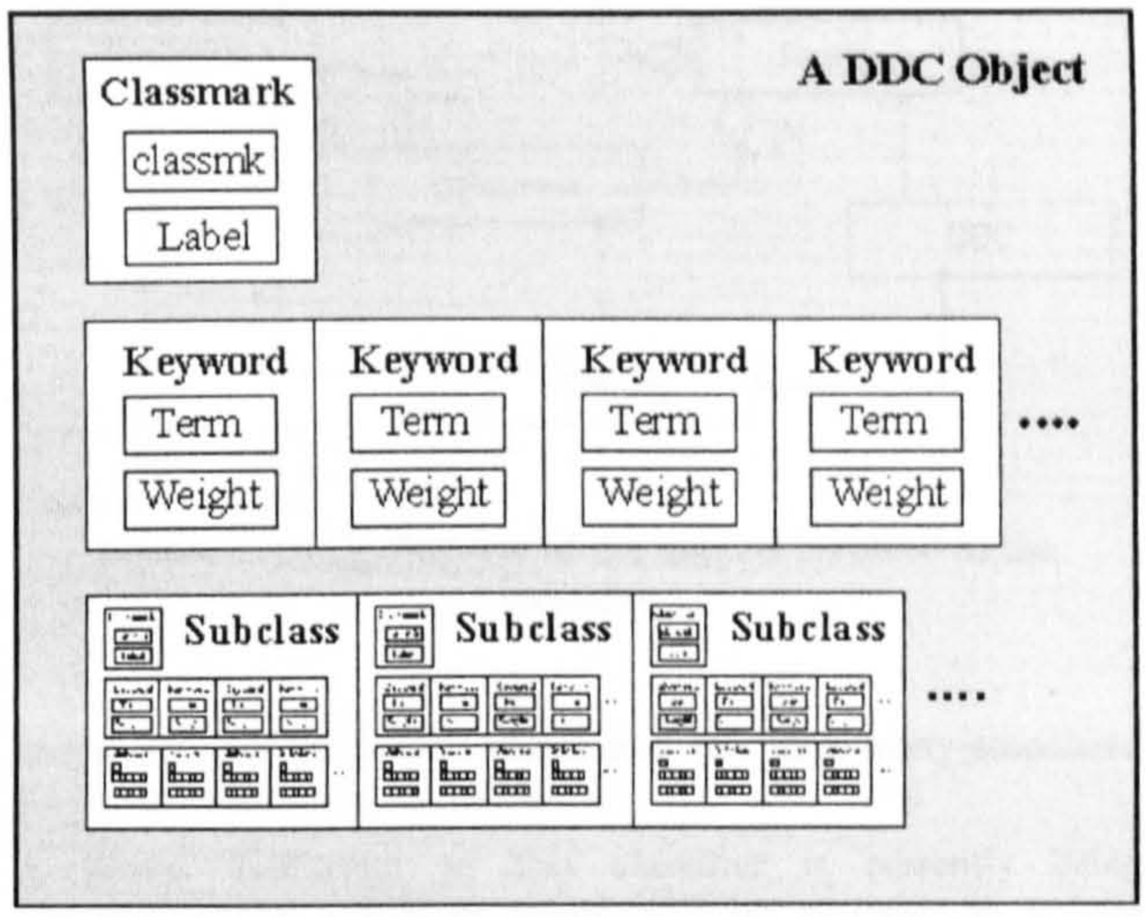


Figure 3. A DDC Object

Each DDC object has a classmark object defining and uniquely identifying the class, a series of keyword objects identical in structure to those defining the document object and a series of subclasses. The keywords form a manually defined class representative of words defining this DDC class. The document keywords are compared with the class representative keywords and if there is a significant measure of similarity (described below) between the two, the document is then compared with any subclasses of this DDC class. The subclasses are DDC objects in themselves representing the next layer of the DDC hierarchy beneath this class. If there are no subclasses (i.e. the class represents a DDC leaf node) and there is a significant measure of similarity, the corresponding classmark is assigned to the classmarks vector in the document object. If the measure of similarity is not significant, the document comparison process will proceed no further through this particular branch of the hierarchy.

000	Generalities
100	Philosophy, paranormal phenomena, psychology
200	Religion
300	Social sciences
400	Language
500	Natural sciences and mathematics
600	Technology (Applied sciences)
700	The arts, Fine and decorative arts
800	Literature (Belles-lettres) and rhetoric
900	Geography, history, and auxiliary disciplines

Figure 4. The top ten DDC classes

The measure of similarity is calculated using the Dice coefficient[13], shown in Figure 5. Each time a word in the document matches a word in the DDC class, the two associated scores are added to a total score ($X \cap Y$). This is then divided by the length of the document (word count) plus the length of the class representative ($X \cup Y$) and the result is multiplied by two. Any result greater than 0.5 is considered significant and the

document comparison process will proceed to any subclasses or the document will be assigned the classmark in the event of a leaf node.

$$2\,\frac{X \cap Y}{X \cup Y}$$

Figure 5. The Dice Coefficient

As the document comparison process works recursively through the DDC object hierarchy, filtering the document down to appropriate leaf nodes, several different branches of the hierarchy can be pursued concurrently. In a Web library multiple classifications are possible - the same 'book' can appear on more than one shelf at a time. The class representatives at the top of the hierarchy contain a broader range of terms than those further down which are more specific. Ambiguous terms can be concealed lower down the hierarchy and considered in context. Figure 6 shows a UML diagram of the main objects involved in the classifier.

The Ace (Automatic Classification Engine) object is made up of a Document object and a Classify object. It first creates a Document object then passes it as a parameter to the Classify object. The Classify object, which co-ordinates the recursive classification process, comprises a Document and is associated with many DDC objects and many classmark objects. A Document object may have many associated Keywords and many Classmarks. A DDC object inherits the Dewey object and may have many associated Keywords but only one Classmark.

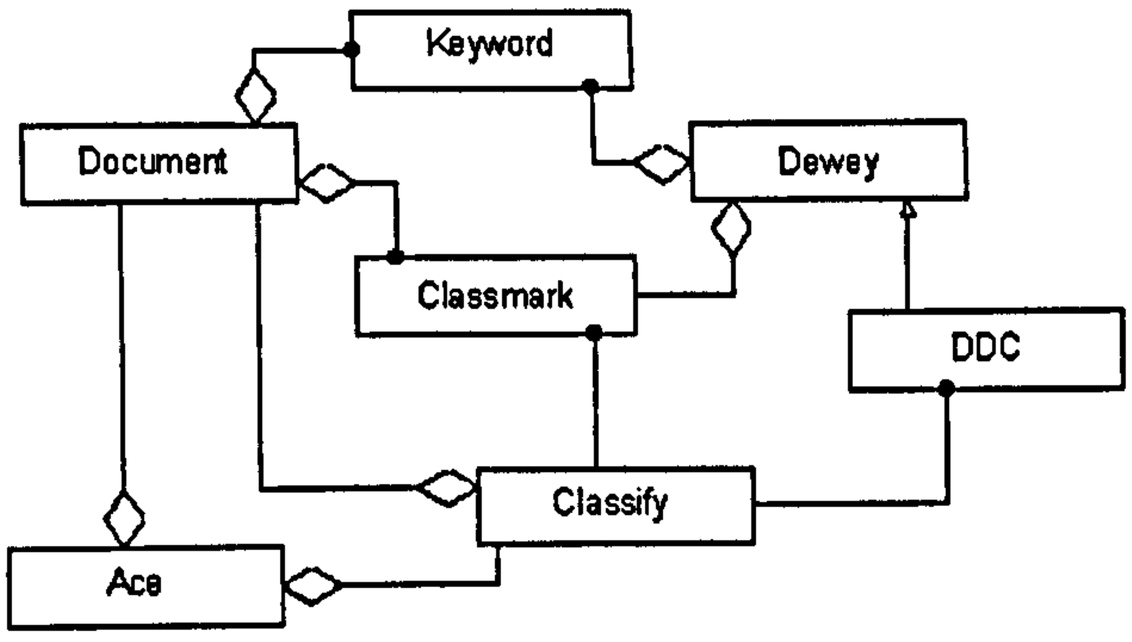


Figure 6. UML diagram of the objects involved in the classifier

A formal evaluation of this classifier is currently being undertaken, the results of which will be available in a later publication. A broad selection of UK Web pages have been manually classified by librarians from the University of Wolverhampton. These classifications are then being compared with classifications assigned by the automatic classifier.

3. METADATA

Automatically generated classmarks can be used to greatly improve the quality of search engine results. The results of a query can be restricted to those belonging to certain areas of the classification scheme. This reduces the tendency to inundate the users with inappropriate results and enables some notion of

	Wolverhampton Core	Description	Purpose	Dublin Core Equivalent
1.	Unique accession number	Number assigned by the system.	Uniquely identifies the resource.	Identifier
2.	Title	Taken from the HTML <TITLE> element.	Usually helps in discerning the subject matter.	Title
3.	URL*	The URL given to the system, used to extract the document for classification.	Indicates the location of the document.	Identifier
4.	Abstract	Either the first 25 words found in the body of the page, or, if present, taken from the Description META tag. (A much more sophisticated abstracting technique could be used here in future implementations).	Provides further clues about the subject matter.	Description
5.	Keywords	Terms found within the document that match terms found within the class representatives of DDC classes found to be appropriate.	Indicate key issues/topics.	Subject
6.	Classmarks	DDC classmarks found to be appropriate as a consequence of the classification process.	Indicate subject area(s).	Subject
7.	Word count	The number of words found on the page, including the title.	Indicates extent, detail, download time.	-
8.	Classification date	The system date when the classification took place (GMT or BST)	Indicates currency of the metadata.	-
9.	Last modified date when classified	Taken from the HTTP Last-modified header. (Gives Not known if equal to the "epoch" - 1st January 1970)	Indicates currency of the information.	Date

Figure 7. Wolverhampton Core elements

context. Such a classifier could also be used to classify queries and so improve the precision of searches.

Other metadata elements describing an HTML page, in addition to the classification classmarks, can be extracted during the process of automatic classification. These elements can then be represented in RDF format which will enable other RDF compliant applications to interpret and share resource descriptions.

3.1 The Wolverhampton Core

Figure 7 lists the metadata elements that are easily extracted during the classification process. These are referred to as Wolverhampton Core. As can be seen, they represent a subset of the 15 Dublin Core elements with one or two additions. These elements are thought to be particularly appropriate for resource descriptions within an automated search engine, providing the ability to uniquely identify the document, state what it is about and when it was last updated. The classification date also shows the currency of the metadata describing the resource.

Maintaining interoperability with a recognized standard such as Dublin Core is clearly an advantage. RDF is a framework on which extensible but interoperable metadata element sets can be defined and interpreted. The following subsection shows how the element set shown in figure 7 can be defined in RDF.

3.2 RDF

The RDF data model is expressed using nodes and arcs diagrams (see the RDF Model and Syntax specification[9]). Figure 8 shows an RDF data model of the elements shown in figure 7.

The diagram shows two instances of RDF container classes. The system is configured to output the two highest scoring classmarks (if more than one is found appropriate), these are ordered with the highest scoring one being presented first so an RDF sequence is appropriate. The keywords may be several in number but are not ordered so an RDF bag is appropriate. In order to express these properties in RDF syntax appropriate RDF schemas (see the RDF Schema Specification[1]) must be identified or defined. RDF schemas contain the type definitions for interpreting RDF statements describing a resource. In order to define the properties shown in figure 7 and 8 it would have been possible to reference the Dublin Core schema directly from within the RDF syntax. However, it was important that our search engine application could differentiate between classmarks and keywords, for example, both of which map on to the Dublin Core Subject property. Although such distinctions could be made using Dublin Core qualifiers[4], the option chosen instead was to define a completely new schema for the Wolverhampton Core that defines those properties that can be mapped on to Dublin Core properties as sub properties e.g.

```
<rdf:Description ID="Classmark">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-
rdf-syntax#Property"/>
  <rdfs:subPropertyOf
resource="http://purl.org/metadata/dublin_core#Sub
ject"/>
  <rdfs:label>Classmark</rdfs:label>
</rdf:Description>
```

This means that an application capable of processing Dublin Core could process this classmark property as if it were a Dublin Core Subject property. The full Wolverhampton Core schema can be found in Appendix A of this paper. This use of RDF demonstrates clearly how the framework can be used to reuse and build upon existing schemas to define extensible element sets for a range of uses. Using existing schemas in this manner encourages interoperability between applications.

Finally, automatically generated RDF can be presented following the property definitions of the Wolverhampton Core schema.

Figure 9 shows an automatically generated RDF description of the Workshop on Organizing Web Space home page. Appendix B shows automatically generated RDF for a range of resources.

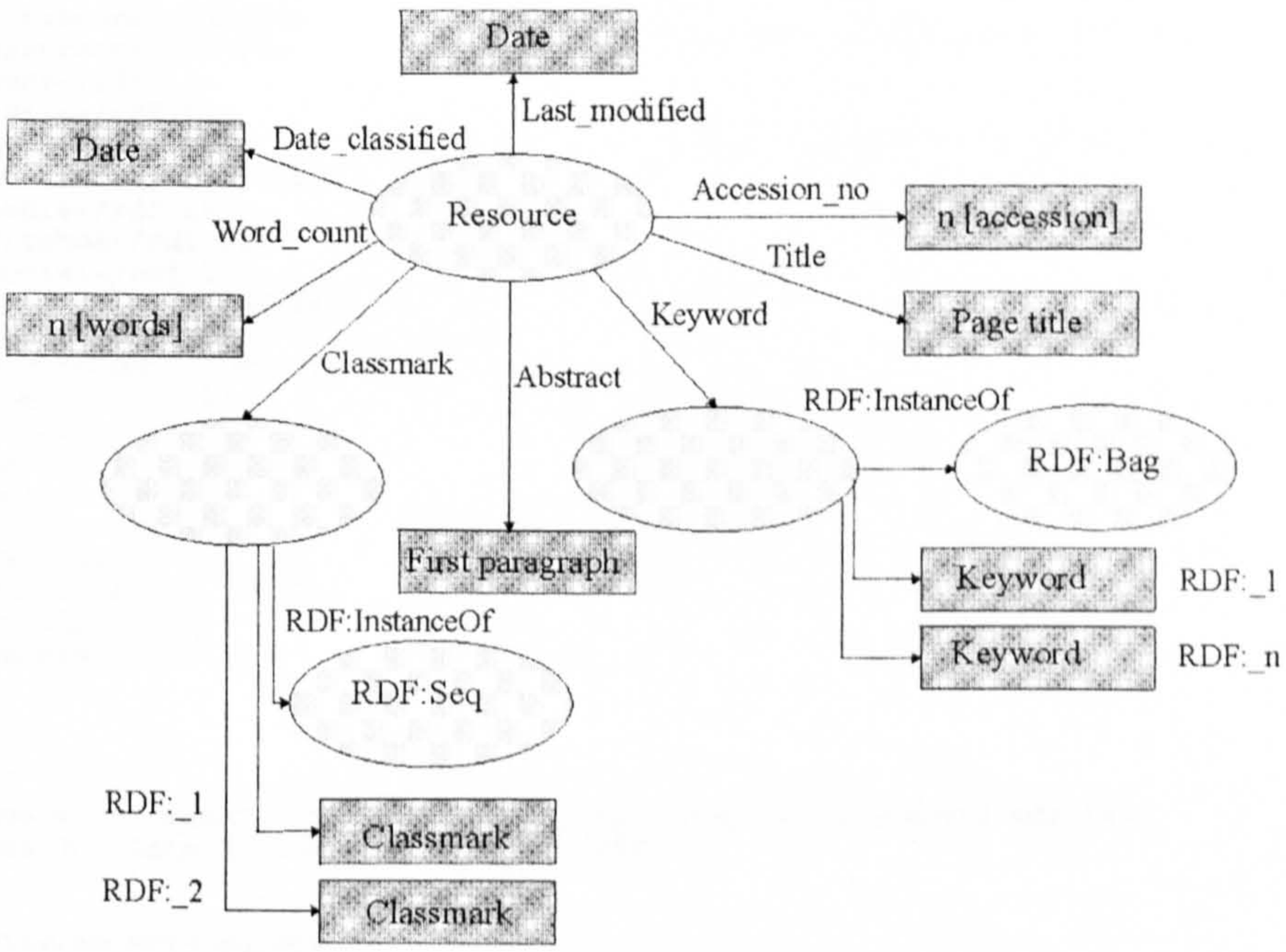


Figure 8. RDF data model

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://www.ccrl.neclab.com/dl99ws/">
                <wc:Accession_no>0</wc:Accession_no>
                <wc:Title>Call for Papers Workshop on Organizing Web Space WOWS</wc:Title>
                <wc:Abstract>Call for Papers Workshop on Organizing Web Space WOWS In
conjunction with ACM Digital Library '99 Radisson Hotel Berkeley CA USA August 14th 1999
Motivation</wc:Abstract>
                <wc:Keyword>
                   <rdf:Bag>
                       <rdf:li>library</rdf:li>
                       <rdf:li>libraries</rdf:li>
                       <rdf:li>ir</rdf:li>
                       <rdf:li>computer</rdf:li>
                       <rdf:li>computers</rdf:li>
                       <rdf:li>hypertext</rdf:li>
                       <rdf:li>ibm</rdf:li>
                       <rdf:li>systems</rdf:li>
                       <rdf:li>communications</rdf:li>
                       <rdf:li>data</rdf:li>
                       <rdf:li>email</rdf:li>
                       <rdf:li>interactions</rdf:li>
                       <rdf:li>analysis</rdf:li>
                       <rdf:li>web</rdf:li>
                       <rdf:li>com</rdf:li>
                       <rdf:li>databases</rdf:li>
                       <rdf:li>program</rdf:li>
                       <rdf:li>engineering</rdf:li>
                       <rdf:li>documentation</rdf:li>
                       <rdf:li>management</rdf:li>
                       <rdf:li>ascii</rdf:li>
                       <rdf:li>search</rdf:li>
                       <rdf:li>file</rdf:li>
                       <rdf:li>retrieval</rdf:li>
                       <rdf:li>school</rdf:li>
                       <rdf:li>classifying</rdf:li>
                       <rdf:li>collection</rdf:li>
                       <rdf:li>structure</rdf:li>
                       <rdf:li>abstract</rdf:li>
                       <rdf:li>user</rdf:li>
                       <rdf:li>users</rdf:li>
                       <rdf:li>access</rdf:li>
                       <rdf:li>university</rdf:li>
                       <rdf:li>media</rdf:li>
                       <rdf:li>Krishna</rdf:li>
                       <rdf:li>social</rdf:li>
                       <rdf:li>communities</rdf:li>
                       <rdf:li>phone</rdf:li>
                       <rdf:li>environment</rdf:li>
                       <rdf:li>postal</rdf:li>
                       <rdf:li>science</rdf:li>
                       <rdf:li>navigation</rdf:li>
                       <rdf:li>space</rdf:li>
                       <rdf:li>paper</rdf:li>
                       <rdf:li>explosive</rdf:li>
                       <rdf:li>photographs</rdf:li>
                       <rdf:li>area</rdf:li>
                       <rdf:li>camera</rdf:li>
                   </rdf:Bag>
                </wc:Keyword>
                <wc:Classmark>
                   <rdf:Seq>
                       <rdf:li>004.6 Interfacing and communications (Computer science)</rdf:li>
                       <rdf:li>005.7   Data in computer systems</rdf:li>
                   </rdf:Seq>
                </wc:Classmark>
                <wc:Word_count>716</wc:Word_count>
                <wc:Classification_date>28-May-99 08:26:32</wc:Classification_date>
                <wc:Last_modified>05-May-99 20:32:19</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

Figure 9. Automatically Generated RDF Metadata

4. CONCLUSIONS

Comprehensive, accurate, reliable, unbiased resource descriptions are clearly essential for improved resource discovery and for facilitating the *Web of Trust*. RDF forms the framework on which appropriate, extensible metadata element sets can be defined and interpreted in a standard notation, but how can the actual resource descriptions be obtained in a comprehensive and reliable manner? Automatic resource description is the obvious answer. Automatic RDF metadata generation, as demonstrated in this paper, enables not only comprehensive, unbiased resource description of any HTML page but also encourages interoperability between applications. RDF enables developers to take a standard element set, or several standard element sets, and extend them to suit their particular domain if required. Such interoperability will be important for the widespread deployment of RDF applications and for improvements in Web coverage and information sharing between Web search engines and information gateways.

5. REFERENCES

[1] Brickley, Guha, Resource Description Framework (RDF) Schema Specification, http://www.w3.org/TR/WD-rdf-schema, Proposed Recommendation 03 March 1999

[2] Burden, The Wolverhampton Web Library (WWLib) http://www.scit.wlv.ac.uk/wwlib/, 1995

[3] Connolly, Bosak, Extensible Markup Language (XML), http://www.w3.org/XML/, October 1998

[4] Dublin Core Metadata Initiative, http://purl.org/dc/, 1999

[5] Fairthorne, The mathematics of classification: Towards Information Retrieval, Butterworths, 1961

[6] Forest Press, OCLC, Dewey Decimal System Home Page, http://www.oclc.org/oclc/fp/index.htm (October 1998)

[7] Good, Speculations Concerning Information Retrieval, Research Report PC-78, IBM Research Center, 1958

[8] Jenkins, Jackson, Burden, Wallis, Automatic Classification of Web Resources using Java and Dewey Decimal Classification, Computer Networks and ISDN Systems, Volume 30 646-648, 1998

[9] Lassila, Swick, Resource Description Framework (RDF) Model and Syntax Specification, http://www.w3.org/TR/PR-rdf-syntax/, Proposed Recommendation 05 January 1999

[10] Lawrence, Giles, Searching the World Wide Web, SCIENCE, Volume 280, April 1998

[11] Resnick, Platform for Internet Content Selection (PICS), http://www.w3.org/PICS/, January 1998

[12] Salton, Automatic Information Organization and Retrieval, McGraw-Hill, New York, 1968

[13] van Rijsbergen, Information Retrieval: Second Edition, Chapter 3, http://www.dcs.glasgow.ac.uk/Keith/Chapter.3/Ch.3.html Butterworths, ISBN 0-408-10775-8, 1981

[14] Weibel, Milller, Dublin Core Metadata, http://purl.oclc.org/metadata/dublin_core/, November 1998

[15] The World Wide Web Consortium, http://www.w3.org 1999

APPENDIX A

The Wolverhampton Core RDF schema.

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
   xmlns:rdfs="http://www.w3.org/TR/WD-rdf-schema#">

<rdf:Description ID="Accession_no">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Identifier"/>
  <rdfs:label>Accession_no</rdfs:label>
  <rdfs:comment>A unique number assigned by the automatic classifier
  that uniquely identifies this resource.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Title">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Title"/>
  <rdfs:label>Title</rdfs:label>
  <rdfs:comment>The title of the resource taken from the HTML TITLE element.
  </rdfs:comment>
</rdf:Description>

<rdf:Description ID="Abstract">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Description"/>
  <rdfs:label>Abstract</rdfs:label>
  <rdfs:comment>This is the first 25 words taken from the BODY of the HTML
  page, or, if present, text taken from the description HTML META tag.
  </rdfs:comment>
</rdf:Description>

<rdf:Description ID="Keyword">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Subject"/>
  <rdfs:label>Keyword</rdfs:label>
  <rdfs:comment> This is a keyword from the document that matched a keyword
  in an appropriate DDC class representative. A number of keywords will
  normally appear in an RDF Bag container.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Classmark">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Subject"/>
  <rdfs:label>Classmark</rdfs:label>
  <rdfs:comment>This is a DDC classmark that has been assigned to the document
  as a result of the automatic classification process. Often two appropriate
  classmarks will be shown in an RDF sequence - the highest scoring one
  appearing first.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Word_count">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:label>Word_count</rdfs:label>
  <rdfs:comment>This is the number of individual words found in the
  resource.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Classification_date">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:label>Classification_date</rdfs:label>
  <rdfs:comment>The date on which the resource was classified.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Last_modified">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Date"/>
  <rdfs:label>Last_modified</rdfs:label>
  <rdfs:comment>The date on which the resource was last modified
  when it was classified.</rdfs:comment>
</rdf:Description>

</rdf:RDF>
```

APPENDIX B

Automatically generated RDF descriptions for a range of HTML pages.

The RSPCA (Royal Society for Protection against Cruelty to Animals)

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://www.rspca.org.uk">
                <wc:Accession_no>0</wc:Accession_no>
                <wc:Title>Welcome to the RSPCA Web Site</wc:Title>
                <wc:Abstract>Royal Society for the Prevention of Cruelty to Animals RSPCA - official
website of the world s largest animal welfare organisation Advice on campaigning pet care careers and how
to help this UK charity</wc:Abstract>
                <wc:Keyword>
                        <rdf:Bag>
                                <rdf:li>society</rdf:li>
                                <rdf:li>education</rdf:li>
                                <rdf:li>welfare</rdf:li>
                                <rdf:li>children</rdf:li>
                                <rdf:li>animal</rdf:li>
                                <rdf:li>animals</rdf:li>
                                <rdf:li>food</rdf:li>
                                <rdf:li>farm</rdf:li>
                                <rdf:li>farming</rdf:li>
                        </rdf:Bag>
                </wc:Keyword>
                <wc:Classmark>
                        <rdf:Seq>
                                <rdf:li>630     Agriculture and Related Technologies</rdf:li>
                                <rdf:li>590     Animals</rdf:li>
                        </rdf:Seq>
                </wc:Classmark>
                <wc:Word_count>0</wc:Word_count>
                <wc:Classification_date>28-May-99 13:46:35</wc:Classification_date>
                <wc:Last_modified>28-May-99 09:56:40</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

The Royal Horticultural Society

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://www.rhs.org.uk">
                <wc:Accession_no>0</wc:Accession_no>
                <wc:Title>null</wc:Title>
                <wc:Abstract>    Welcome to the Royal Horticultural Society's website Chelsea
Flower Show The most inspiring event on the horticultural calendar and the perfect way to</wc:Abstract>
                <wc:Keyword>
                        <rdf:Bag>
                                <rdf:li>library</rdf:li>
                                <rdf:li>database</rdf:li>
                                <rdf:li>internet</rdf:li>
                                <rdf:li>buy</rdf:li>
                                <rdf:li>search</rdf:li>
                                <rdf:li>directory</rdf:li>
                                <rdf:li>public</rdf:li>
                                <rdf:li>society</rdf:li>
                                <rdf:li>Women</rdf:li>
                                <rdf:li>plants</rdf:li>
                                <rdf:li>flowers</rdf:li>
                                <rdf:li>flower</rdf:li>
                                <rdf:li>gardening</rdf:li>
                                <rdf:li>gardeners</rdf:li>
                                <rdf:li>plant</rdf:li>
                                <rdf:li>fruit</rdf:li>
                                <rdf:li>forest</rdf:li>
                                <rdf:li>garden</rdf:li>
                                <rdf:li>gardens</rdf:li>
                                <rdf:li>horticulture</rdf:li>
                                <rdf:li>disease</rdf:li>
                        </rdf:Bag>
                </wc:Keyword>
```

```
                <wc:Classmark>
                        <rdf:Seq>
                                <rdf:li>580     Plants</rdf:li>
                                <rdf:li>630     Agriculture and Related Technologies</rdf:li>
                        </rdf:Seq>
                </wc:Classmark>
                <wc:Word_count>319</wc:Word_count>
                <wc:Classification_date>28-May-99 13:48:07</wc:Classification_date>
                <wc:Last_modified>Not known</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

The University of Wolverhampton

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://www.wlv.ac.uk">
                <wc:Accession_no>0</wc:Accession_no>
                <wc:Title>University of Wolverhampton Home Page</wc:Title>
                <wc:Abstract>About the University For Prospective Students Academic Schools For Students
Contact Us For Staff Quick Links For Alumni News Events Learning at Work Day Quiz</wc:Abstract>
                <wc:Keyword>
                        <rdf:Bag>
                                <rdf:li>newsletter</rdf:li>
                                <rdf:li>news</rdf:li>
                                <rdf:li>press</rdf:li>
                                <rdf:li>university</rdf:li>
                                <rdf:li>schools</rdf:li>
                                <rdf:li>management</rdf:li>
                                <rdf:li>art</rdf:li>
                        </rdf:Bag>
                </wc:Keyword>
                <wc:Classmark>
                        <rdf:Seq>
                                <rdf:li>070.1   Documentary media, educational media, news media</rdf:li>
                                <rdf:li>370     Education</rdf:li>
                        </rdf:Seq>
                </wc:Classmark>
                <wc:Word_count>85</wc:Word_count>
                <wc:Classification_date>28-May-99 13:56:32</wc:Classification_date>
                <wc:Last_modified>28-May-99 12:17:34</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

Automatic RDF Metadata Generation for Resource Discovery

Charlotte Jenkins, Mike Jackson, Peter Burden, Jon Wallis
School of Computing & IT
University of Wolverhampton
Wulfruna Street, Wolverhampton
WV1 1SB, UK

Abstract

Automatic metadata generation may provide a solution to the problem of inconsistent, unreliable metadata describing resources on the Web. The Resource Description Framework (RDF [10]) provides a domain-neutral foundation on which extensible element sets can be defined and expressed in a standard notation. This paper describes how an automatic classifier, that classifies HTML documents according to Dewey Decimal Classification (DDC [8]), can be used to extract context sensitive metadata which is then represented using RDF. The process of automatic classification is described and an appropriate metadata element set is identified comprising those elements that can be extracted during classification. An RDF data model and an RDF schema are defined representing the element set and the classifier is configured to output the elements in RDF syntax according to the defined schema.

Keywords: RDF, metadata, classification.

---

1. Introduction

A major problem facing tools for information resource discovery on the Web is the lack of a mechanism for resource description within the Web's architecture. There are now said to be in excess of 320 million individually accessible objects on the Web [5]. There is no one accurate, reliable, up-to-date, comprehensive method of finding out what each one of these objects is, what type of resource it is, what the subject matter is and so on, without accessing and analysing each one individually. This is a problem, not only for resource discovery, but also for content rating where illicit material is concerned. The World Wide Web Consortium (W3C [13]) has introduced the Resource Description Framework (RDF), in an attempt to produce a standard language for *machine-understandable* descriptions of resources on the Web. RDF is intended to support resource descriptions for resource discovery and also for rights management, privacy preferences, content ratings (PICS [9]), evaluation and classification. RDF is seen as the framework for producing a *Web of trust* where the content of each individually accessible object is well described in a format that is extensible yet universally understood. RDF may enable search engines and other tools for resource discovery to exchange and share metadata. This paper is concerned with the automatic generation of metadata in RDF format for use in describing HTML documents for the purposes of resource discovery.

Various attempts have been made to introduce embedded metadata into HTML documents, most notably through the use of the HTML META tag and embedded Dublin Core [12]. It is also now possible to include an embedded RDF description of a document. The problem with such techniques is that they are not compulsory so many authors still choose not to include meta information. M. Marchiori, in his paper entitled *The Limits of Web Metadata and Beyond* [7], addresses this issue by proposing a scheme that involves *back-propagating* meta information from pages with known metadata to those that are linked from it. An alternative method of automatically generating metadata is to use an automatic classifier. The automatic classifier described in this paper works by comparing terms found within documents with manually defined clusters of terms representing the nodes of a classification hierarchy (DDC). This process results in the identification of other useful metadata such as the document title, keywords, abstract and word count in addition to the classification classmarks. An RDF schema has been defined for representing this metadata and the process by which it is extracted and represented in RDF is described.

---

2. Automatic Classification

The automatic classifier [3] described below has been designed and developed for use as an automated component of a distributed automated search engine. The use of automatic classification within an automated search engine is quite unusual - commonly automated search engines (such as AltaVista) are huge indexes and classified tools (such as Yahoo and Galaxy) require some degree of manual intervention, typically in specifying the classification category and other such meta information. It has been observed [6] that, classified tools, although often hopelessly incomplete and out-of-date because of the lack of automation, are less likely to inundate users with irrelevant information. Automatic resource discovery combined with automatic full text indexing is faster and more comprehensive than manual classification but much less accurate. It is hoped that the use of automatic classification will combine the advantages of both approaches resulting in an accurate, comprehensive, up-to-date, well classified, automated search engine. Documents sharing the same subject matter will be automatically clustered together under the same classification classmarks and therefore will be retrieved together more easily.

The automatic classifier classifies documents according to DDC. DDC is considered appropriate because it is a universal classification scheme covering all subject areas and geographically global information. It is familiar to anyone accustomed to using a library and has multilingual scope. The hierarchical nature enables the users of a search engine to refine their search from rough classifications to increasingly more accurate ones.

The automatic classifier is an object oriented system, written in Java, that retrieves HTML documents from given URLs, analyses the contents and assigns appropriate DDC classmarks. A hierarchy of Java classes is used to model the DDC classification hierarchy. Documents are filtered through this hierarchy according to which *class representatives* (manually defined terms representing each DDC class) best match the document's contents. Each term found within the document is given an associated weight which is greater if the term is found in the title or a heading element. Terms found within the *keywords* or *description* elements of existing META tags are also stored with significant associated weight. Terms also acquire more weight the more often they occur. These weighted terms are then compared with the manually defined terms representing DDC classes. Initially the document is compared with

the top ten DDC classes shown in Figure 1.

000 Generalities
100 Philosophy, paranormal phenomena, psychology
200 Religion
300 Social sciences
400 Language
500 Natural sciences and mathematics
600 Technology (Applied sciences)
700 The arts, Fine and decorative arts
800 Literature (Belles-lettres) and rhetoric
900 Geography, history, and auxiliary disciplines

Figure 1. The ten DDC classes

If a significant match is found between the document and a DDC class, the document is then compared with subclasses of that DDC class. This comparison process continues recursively through the hierarchy until significant matches with leaf nodes are found, the classmarks of which are assigned to the document.

To illustrate this process more clearly, figure 2 shows a document object which comprises:

- an accession number that is used to uniquely identify the document
- a series of keyword objects, each one representing a term found within the document, with an associated weight depending on where it was found within the document and how frequently it occurs (note, ALL found terms are stored in this manner)
- a series of classmark objects, each one comprising the actual classmark together with a textual label e.g. *303.483 Development of science and technology*. Appropriate classmark objects are assigned here when the keywords match significantly with the keywords of DDC objects that have no subclasses (see figure 3) i.e. leaf nodes in the hierarchy.

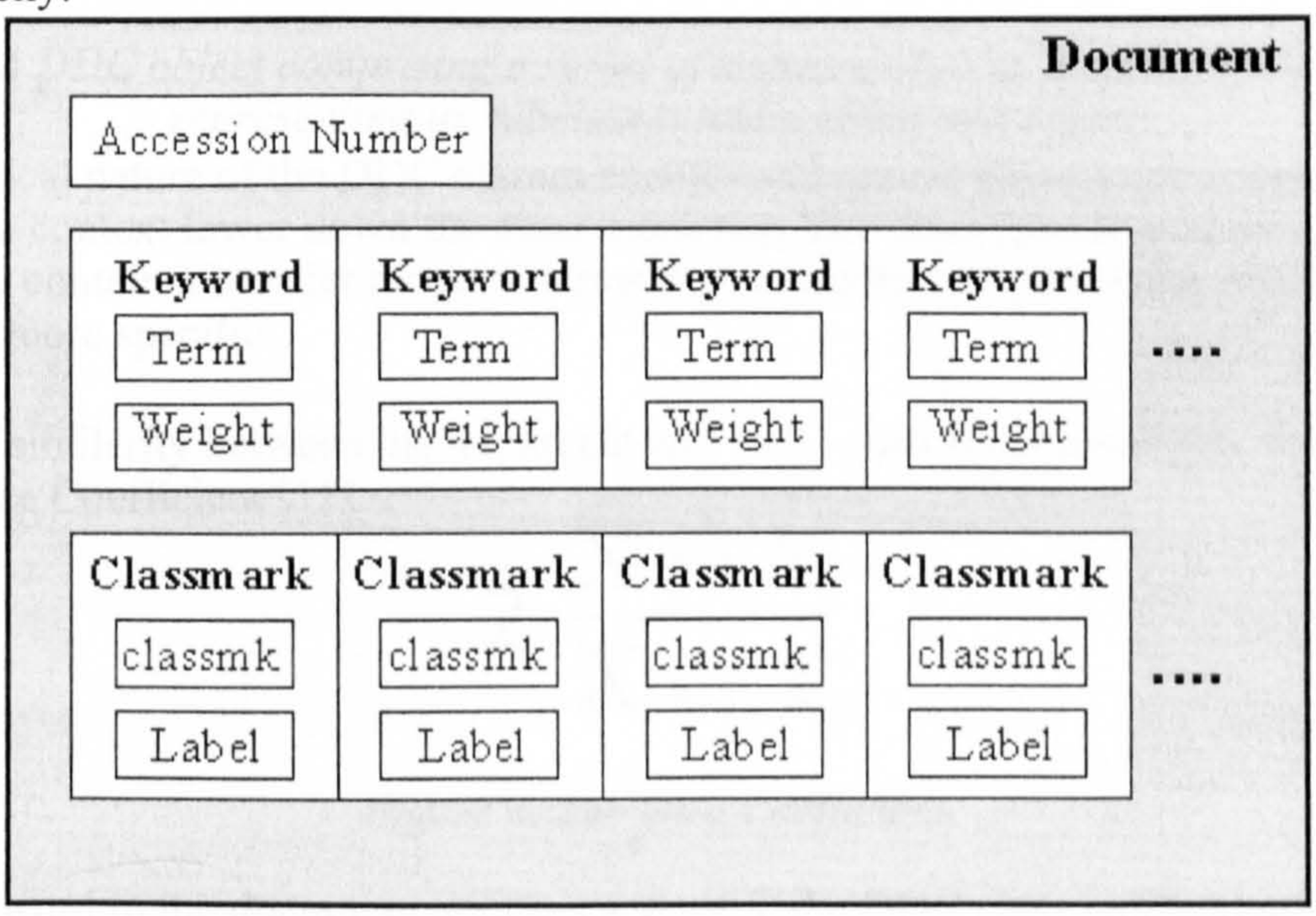


Figure 2. A document object comprising a series of keyword objects and a series of classmark objects

Figure 3 shows a DDC object which comprises:

- a series of keyword objects, identical in structure to the document keywords but comprising manually defined terms representing this particular DDC class
- a series of subclasses - each of which is itself a DDC class representing the next layer of the hierarchy beneath this class. Leaf nodes obviously have no subclasses
- a classmark object defining and uniquely identifying this class. If the keywords of this class match significantly with the keywords of a document object and there are no subclasses, this classmark object is assigned to the document.

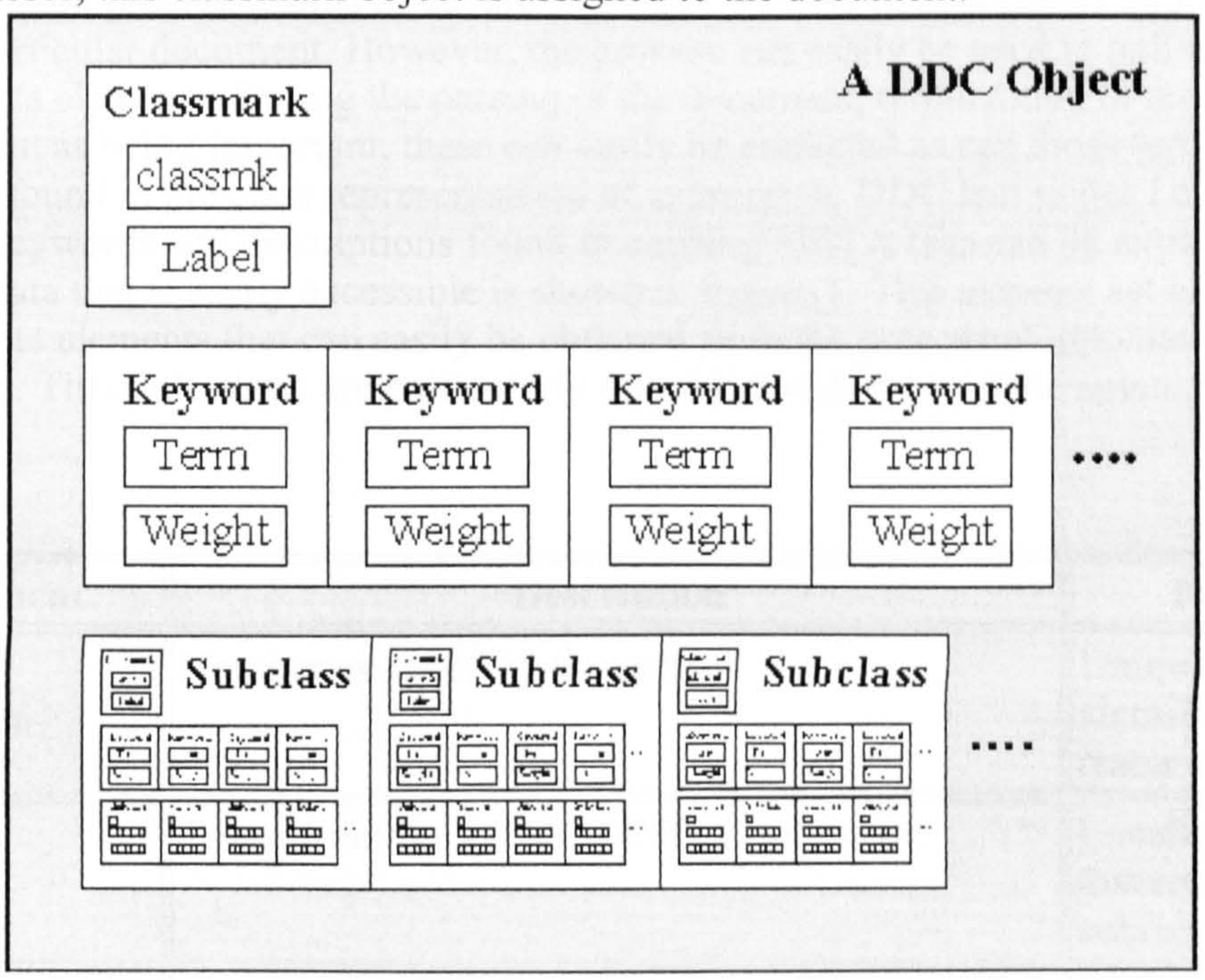


Figure 3. A DDC object comprising a series of keyword objects, a series of DDC objects representing its subclasses and a classmark object

The hierarchical nature of the DDC classes enables ambiguous terms to be concealed and considered in context lower down the class hierarchy. The class representatives at the top of the hierarchy contain a broader range of terms than those nearer the bottom which are more detailed and more specific.

Measures of similarity between the document and DDC *class representatives* are calculated using the Dice Coefficient [11]:

$$2\,\frac{X \cap Y}{X \cup Y}$$

Figure 4. The Dice Coefficient

Each time a word in the document matches a word in the DDC class representative, the two associated weights are added to a total score (x intersection y). This score is then divided by the sum of the number of keywords in the document and the number of keywords in the class representative (x union y) and the result is multiplied by 2. Any result greater than 0.5 is considered significant and the document will proceed to be compared with any subclasses or be assigned the classmark if there are no subclasses. If the score is not significant, the comparison process will proceed no further through this branch of the hierarchy.

The comparison process may proceed through several unrelated branches of the hierarchy for as long as significant matches are found. In a Web library multiple classifications are appropriate - the same book can appear on several different shelves.

2.1 Metadata Elements

The classification process results in the production of a series of classmarks appropriate to describe a particular document. However, the process can easily be used to pull out various other metadata elements. During the parsing of the document, terms found in the title element are singled out as being important, these can easily be extracted as can those terms which match those found in the class representatives of appropriate DDC leaf nodes i.e. significant keywords. Keywords and descriptions found in existing META tags can be extracted. Other useful metadata that is easily accessible is shown in Figure 5. This element set is based on those metadata elements that can easily be obtained from the process of automatic classification. These elements are particularly suited to the domain of the automated search engine.

	Element	Description	Purpose
1.	Unique accession number	Number assigned by the system.	Uniquely identifies the resource.
2.	Title	Taken from the HTML <TITLE> element.	Usually helps in discerning the subject matter.
3.	URL*	The URL given to the system, used to extract the document for classification.	Indicates the location of the document.
4.	Abstract	Either the first 25 words found in the body of the page, or, if present, taken from the *Description* META tag. (A much more sophisticated abstracting technique could be used here in future implementations).	Provides further clues about the subject matter.
5.	Keywords	Terms found within the document that match terms found within the class representatives of DDC classes found to be appropriate.	Indicate key issues/topics.
6.	Classmarks	DDC classmarks found to be appropriate as a consequence of the classification process.	Indicate subject area(s).
7.	Word count	The number of words found on the page, including the title.	Indicates extent, detail, download time.
8.	Classification date	The system date when the classification took place (GMT or BST)	Indicates currency of the metadata.
9.	Last modified date when classified	Taken from the HTTP Last-modified header. (Gives *Not known* if equal to the "epoch" - 1st January 1970)	Indicates currency of the information.

* The classifier only handles individual HTML documents so the URL, not URI, is appropriate. The URL is not used as an identifier within the search engine because it is possible for the same page to have more than one URL; this is one of the causes of repetitions in automated search engine results.

Figure 5. An appropriate metadata element set - The " Wolverhampton Core "

It is thought that these elements (Wolverhampton Core) are sufficient to uniquely identify the document, state where it can be found, provide a good indication of the subject matter and of how current both the actual information and its metadata are.

The most well known and well used metadata element set for resource discovery is Dublin Core [12]. Compliance with a recognised standard is advisable because it encourages interoperability and consistency between applications. Dublin Core has evolved from the Digital Library community and consequently not all of its elements are as well suited to the automated search engine domain as those defined in figure 5. There is, however a significant overlap and none of the Dublin Core elements are compulsory. RDF enables developers to tailor an element set to suit their application while still reusing appropriate standard elements defined elsewhere (see section 3).

Figure 6 compares the fifteen elements of Dublin Core with the elements defined in figure 5.

	Dublin Core Elements	**Equivalent Wolverhampton Core Elements**
1.	Title	Title
2.	Creator	-
3.	Subject	Keywords + Classmarks
4.	Description	Abstract
5.	Publisher	-
6.	Contributor	-
7.	Date	Last modified when classified
8.	Type	-
9.	Format	-
10.	Identifier	Accession number + URL
11.	Source	-
12.	Language	-
13.	Relation	-
14.	Coverage	-
15.	Rights	-
16.	-	Date Classified
17.	-	Word count

Figure 6. Comparison between Dublin Core and the Wolverhampton Core element sets.

It can be observed that most of the *Wolverhampton Core* elements have a Dublin Core equivalent. The implications of this comparison are discussed again in the later section (3.2) on RDF schema definition. It is thought that the specified Wolverhampton Core elements

represent an appropriate subset of Dublin Core (with one or two additions) that is suited to the requirements of an automated search engine.

Once the necessary metadata elements have been identified they can then be represented in RDF.

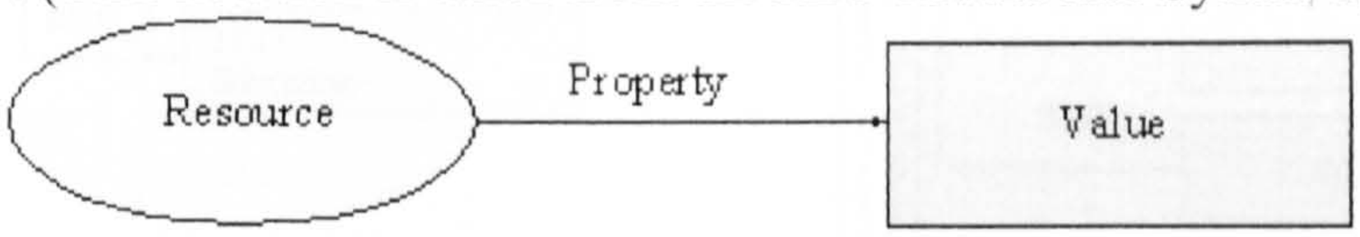
---

3. Resource Description Framework (RDF)

Three things are required in order to generate RDF statements about a resource: a data model, a schema and the actual representation in XML (eXtensible Markup Language [2]) syntax. Several RDF schemas might actually be involved; schemas are required for the interpretation of RDF statements. The following three subsections explain how the metadata elements shown in figure 5 can be represented by an RDF data model, defined using an RDF schema and, most importantly, automatically generated in RDF/XML syntax.

3.1 RDF Data Model

The RDF data model is expressed using directed labelled graphs (or "nodes and arcs" diagrams) which identify the properties and property values associated with a resource as shown in figure 7. (This notation is taken from the RDF Model and Syntax Specification [4]).



e.g.

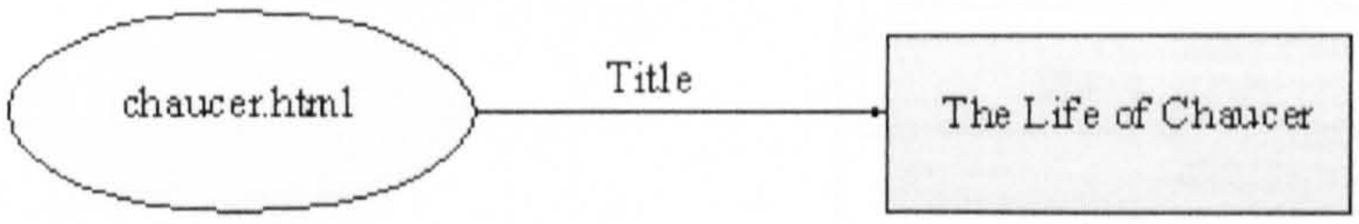


Figure 7. Data Model notation showing an RDF statement; a resource, a named property and the value of that property.

In RDF a resource may be a simple Web page, part of a simple Web page, a collection of pages or a whole Web site. The automatic metadata generator described in this paper generates descriptions of individual HTML pages.

Figure 8 shows how the element set in Figure 5 would be represented for the HTML page at http://www.scit.wlv.ac.uk/index.html

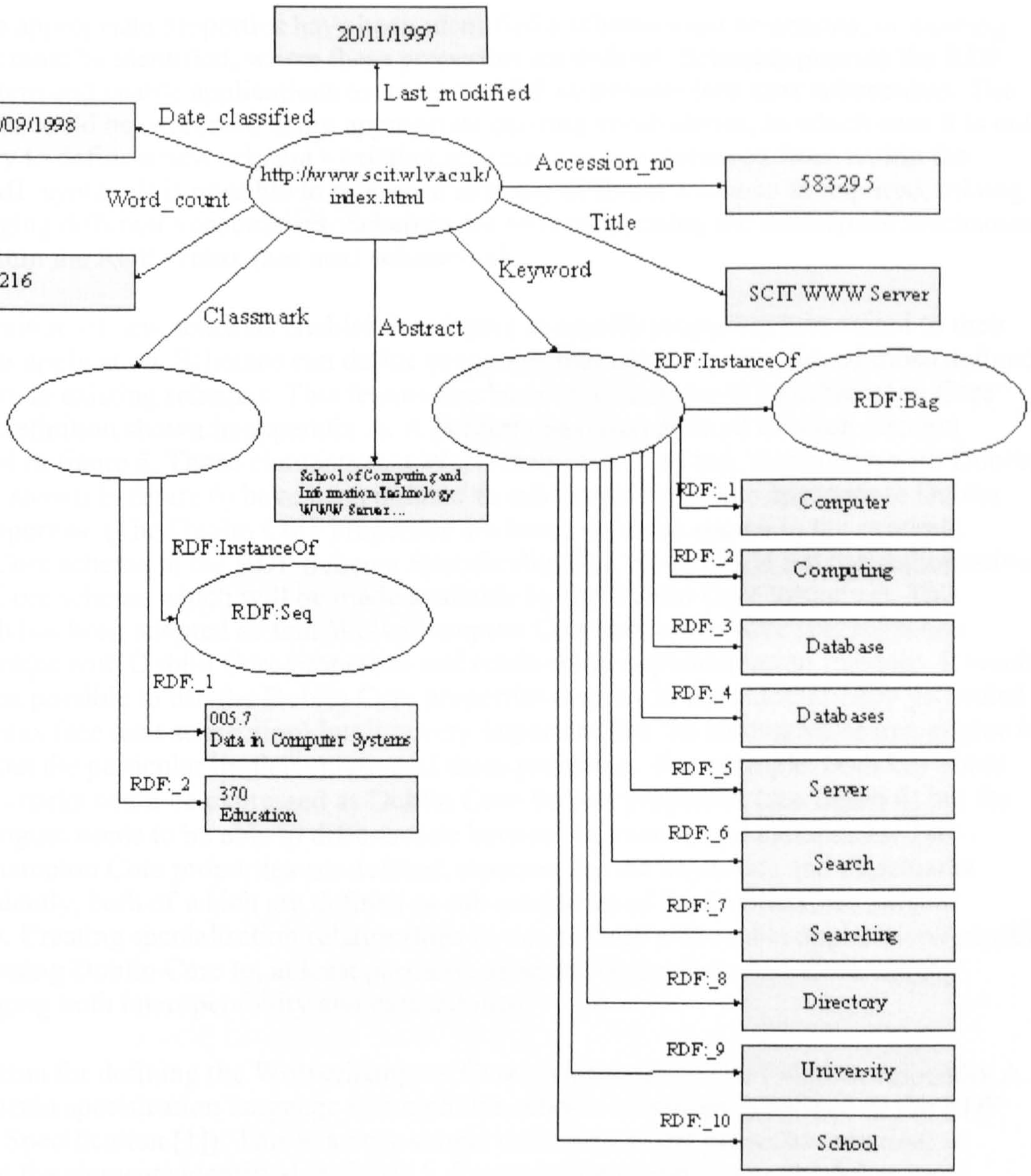


Figure 8. An RDF Data Model using the elements from Figure 5.

The model shows two RDF containers - one a bag of keywords and the other a sequence of classmarks. The classification process will usually result in the identification of several keywords within the document but the order in which they are presented is insignificant so a bag is appropriate. A better method of representing the keywords would be to use a *Set* where no duplicates would be permitted, however, RDF does not define a *Set* because there is no defined enforcement mechanism in the event of violation. The classmarks are ordered by the classifier according to which scored the highest measure of similarity and so these are represented as an ordered sequence. The classmarks would be better represented by an ordered collection class where no duplicates would be allowed. Further work layered on the RDF core may define such enforcement mechanisms.

3.2 RDF Schema Definition

Once the appropriate properties have been identified a schema must be created, or existing schemas must be identified, where these properties are defined. Schemas provide the RDF type system and enable applications to interpret RDF statements (see next subsection). The properties could be expressed using appropriate existing vocabularies, in which case it is not necessary to define a new schema - existing schemas can be referenced from within the RDF/XML syntax. It is possible to reference as many different schemas as required, mixing and merging different vocabularies. Schemas are referenced using the namespace mechanism from within the RDF syntax (see next subsection).

The definition of new schemas enables developers to specify properties best suited to their particular application. Schemas can define properties that are sub-properties of those defined elsewhere in existing schemas. This feature has been utilised in the Wolverhampton Core schema definition shown in appendix A. A property has been defined for each element identified in figure 5. Those elements that Wolverhampton Core has in common with Dublin Core (as shown in figure 6) have been defined as sub-properties of the appropriate Dublin Core properties. (The Dublin Core properties are based on those shown in the example Dublin Core schema in the RDF Schema Specification [1]. Note, this is not the authoritative Dublin Core schema which will be made available by the Dublin Core Initiative). This approach has been adopted so that Wolverhampton Core properties have *specialisation relationships* with Dublin Core properties and retain some implementation freedom. It would have been possible to use the Dublin Core properties directly in the automatically generated RDF syntax (see next subsection) but it is very important that the automated search engine is clear about the particular implementation of these properties. For example, both keywords and classmarks could be expressed as Dublin Core *Subject* properties (see figure 6) but the search engine needs to be able to differentiate between keywords and classmarks. Two Wolverhampton Core properties are defined, representing the keywords and classmarks independently, both of which are defined as sub-properties of the Dublin Core *Subject* property. Creating specialisation relationships in this manner will enable applications capable of processing Dublin Core to, at least partially, interpret Wolverhampton Core thereby encouraging both interoperability and extensibility.

The schema for defining the Wolverhampton Core element set can be found in Appendix A. (The schema specification language in which this schema is written is defined in the RDF Schema Specification [1]). This is a very simple definition of the properties required to represent the elements identified in figure 5. Future implementations could define new classes and declare constraints on the properties.

3.3 RDF Syntax

The following shows the RDF representation of the data model shown in figure 8. Appendix B shows automatically generated RDF for a series of test URLs. (The RDF/XML syntax used here is described in The RDF Model and Syntax Specification [4].)

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
     <rdf:Description about="http://www.scit.wlv.ac.uk/">
          <wc:Accession_no>583295</wc:Accession_no>
          <wc:Title>SCIT WWW Server</wc:Title>
          <wc:Abstract>
               School of Computing and Information Technology
```

```
        WWW server General Information University of
        Wolverhampton School of Computing and Information
        Technology home page Wolverhampton and surrounding
        areas
</wc:Abstract>
<wc:Keyword>
        <rdf:Bag>
                <rdf:li>computer</rdf:li>
                <rdf:li>computing</rdf:li>
                <rdf:li>database</rdf:li>
                <rdf:li>databases</rdf:li>
                <rdf:li>server</rdf:li>
                <rdf:li>search</rdf:li>
                <rdf:li>searching</rdf:li>
                <rdf:li>directory</rdf:li>
                <rdf:li>university</rdf:li>
                <rdf:li>school</rdf:li>
        </rdf:Bag>
</wc:Keyword>
<wc:Classmark>
        <rdf:Bag>
                <rdf:li>005.7 Data in computer systems</rdf:li>
                <rdf:li>370   Education</rdf:li>
        </rdf:Bag>
</wc:Classmark>
<wc:Word_count>216</wc:Word_count>
<wc:Last_modified>20/11/1997</wc:Last_modified>
<wc:Classification_date>11/09/1998</wc:Classification_date>
    </rdf:Description>
</rdf:RDF>
```

Note that there are two XML namespace definitions (xmlns) at the top of this piece of RDF. The first one identifies the location of the RDF syntax specification and the second one identifies the location of the Wolverhampton Core (wc) schema where the property types specified within this RDF description are defined. This wc schema is shown in Appendix A.

W3C and the Dublin Core Initiative recommend the use of ISO 8601 Date format. This has not been implemented in this instance because the automatic metadata generator is to be deployed as part of a UK search engine where dates will be required in UK format.

If the classification process should fail, i.e. no significant measures of similarity are found, other elements such as the title, abstract, word count and dates should still be identified.

---

4. Conclusions

Although it is envisaged that the editing tools of the future will encourage the inclusion of RDF meta information, the current situation, where some authors choose not to include any metadata, is likely to continue to some extent. It is very difficult to automate resource description but it would be impossible to describe everything on the Web manually. Automatic metadata generation would appear to be an essential pre-requisite for widespread deployment of RDF based applications. The *Web of trust* must attempt to be comprehensive because a Web that is partially trust worthy offers little advantage over one that cannot be

trusted at all, especially where content rating is concerned.

The automatic metadata generator described in this paper enables an RDF description to be associated with any HTML page, regardless of when it was created and by which editing tool. RDF has enabled the specification of a metadata element set that is tailored to suit an automated search engine but strongly related to a standard, digital library element set, Dublin Core. The ability to create specialisation relationships with appropriate Dublin Core properties increases the potential for interoperability - any application capable of processing Dublin Core will be capable of processing most of the defined Wolverhampton Core properties because they are defined as sub-properties of Dublin Core properties. Such interoperability will encourage information sharing which will improve comprehensive Web coverage; if search engines can process the same standard syntax, they will be able to exchange metadata and integrate their results. Some subject-specific classified directories are known to be attempting to share information through the use of RDF already; information sharing between automated search engines has even greater potential.

References

1. D. Brickley, R. Guha, A. Layman, *Resource Description Framework (RDF) Schema Specification*, http://www.w3.org/TR/WD-rdf-schema (Working Draft - October 1998)
2. D. Connolly, J. Bosak, *Extensible Markup Language (XML)*, http://www.w3.org/XML/, (October 1998)
3. C. Jenkins, M. Jackson, P. Burden, J. Wallis, *Automatic Classification of Web Resources using Java and Dewey Decimal Classification*, Computer Networks and ISDN Systems, Volume 30 646-648 (1998)
4. O. Lassila, R. Swick, *Resource Description Framework (RDF) Model and Syntax Specification*, http://www.w3.org/TR/WD-rdf-syntax (Working Draft - October 1998)
5. S. Lawrence, C. L. Giles, *Searching the World Wide Web*, SCIENCE, Volume 280 (April 1998)
6. L. Lindop, M. Sriskandarajah, M. Williams, M. Bracken, M. Cadden, A. Dabbs, W. Gallagher, *Catching Sites*, PC Magazine 6 (2) (February 1997)
7. M. Marchiori, *The Limits of Web metadata and beyond*, Computer Networks and ISDN Systems, Volume 30 1-9, (1998)
8. OCLC Forest Press, *Dewey Decimal System Home Page*, http://www.oclc.org/oclc/fp/index.htm (October 1998)
9. P. Resnick, *Platform for Internet Content Selection (PICS)*, http://www.w3.org/PICS/ (January 1998)
10. R. Swick, E. Miller, B. Schloss, D. Singer, *Resource Description Framework (RDF)*, http://www.w3.org/RDF/ (October 1998)
11. R. C. J. van Rijsbergen, *Information Retrieval: Second Edition*, Chapter 3, http://www.dcs.glasgow.ac.uk/Keith/Chapter.3/Ch.3.html Butterworths, ISBN 0-408-10775-8, 1981
12. S. Weibel, E. Miller, *Dublin Core Metadata*, http://purl.oclc.org/metadata/dublin_core/ (November 1998)
13. The World Wide Web Consortium, http://www.w3.org (October 1998)

Appendix A

Below is the RDF schema for the Wolverhampton Core (wc) element set referred to in figures 5 and 8. Note that the URL is not specified as a separate property because it is always noted in the `<rdf:Description about="http://...">` statement.

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
   xmlns:rdfs="http://www.w3.org/TR/WD-rdf-schema#">

<rdf:Description ID="Accession_no">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Identifier"/>
  <rdfs:label>Accession_no</rdfs:label>
  <rdfs:comment>A unique number assigned by the automatic classifier
  that uniquely identifies this resource.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Title">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Title"/>
  <rdfs:label>Title</rdfs:label>
  <rdfs:comment>The title of the resource taken from the HTML TITLE element.
  </rdfs:comment>
</rdf:Description>

<rdf:Description ID="Abstract">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Description"/>
  <rdfs:label>Abstract</rdfs:label>
  <rdfs:comment>This is the first 25 words taken from the BODY of the HTML
  page, or, if present, text taken from the description HTML META tag.
  </rdfs:comment>
</rdf:Description>

<rdf:Description ID="Keyword">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Subject"/>
  <rdfs:label>Keyword</rdfs:label>
  <rdfs:comment> This is a keyword from the document that matched a keyword
  in an appropriate DDC class representative. A number of keywords will
  normally appear in an RDF Bag container.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Classmark">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Subject"/>
  <rdfs:label>Classmark</rdfs:label>
  <rdfs:comment>This is a DDC classmark that has been assigned to the document
  as a result of the automatic classification process. Often two appropriate
  classmarks will be shown in an RDF sequence - the highest scoring one
  appearing first.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Word_count">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:label>Word_count</rdfs:label>
  <rdfs:comment>This is the number of individual words found in the
  resource.</rdfs:comment>
</rdf:Description>
```

```
<rdf:Description ID="Classification_date">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:label>Classification_date</rdfs:label>
  <rdfs:comment>The date on which the resource was classified.</rdfs:comment>
</rdf:Description>

<rdf:Description ID="Last_modified">
  <rdf:type rdf:resource="http://www.w3.org/TR/WD-rdf-syntax#Property"/>
  <rdfs:subPropertyOf resource="http://purl.org/metadata/dublin_core#Date"/>
  <rdfs:label>Last_modified</rdfs:label>
  <rdfs:comment>The date on which the resource was last modified
  when it was classified.</rdfs:comment>
</rdf:Description>
```

```
</rdf:RDF>
```

Appendix B

The following RDF descriptions have been automatically generated. The automatic metadata generator is a Java program that retrieves HTML pages from given URLs and automatically analyses and classifies them according to DDC (see section 2). The DDC classmarks along with other accessible metadata elements (see figure 5) are then represented in RDF using the Wolverhampton Core (wc) schema (see Appendix A). The example pages have been selected from the top of a random range of Yahoo categories as indicated. Note that the accession number is not set in the following examples because the program is running as a stand alone application and not within the context of the search engine.

Yahoo - Home : Social Science : Psychology

http://dir.yahoo.com/Social_Science/Psychology/Education

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://www-nmcp.med.navy.mil/psychology/I1.htm">
                <wc:Accession_no>0</wc:Accession_no>
                <wc:Title>I1</wc:Title>
                <wc:Abstract>Psychology Department Home Page Clinical Psychology
                Internship Since 1990 the Psychology Department has offered a predoc
                clinical psychology internship fully accredited by the American Psyc
                <wc:Keyword>
                        <rdf:Bag>
                                <rdf:li>psychology</rdf:li>
                                <rdf:li>psychological</rdf:li>
                                <rdf:li>association</rdf:li>
                                <rdf:li>adult</rdf:li>
                                <rdf:li>training</rdf:li>
                                <rdf:li>leadership</rdf:li>
```

```
                        <rdf:li>American</rdf:li>
                        <rdf:li>navy</rdf:li>
                        <rdf:li>naval</rdf:li>
                </rdf:Bag>
</wc:Keyword>
<wc:Classmark>
                <rdf:Seq>
                        <rdf:li>350     Public Administration and Military S
                        <rdf:li>158     Applied psychology</rdf:li>
                </rdf:Seq>
</wc:Classmark>
<wc:Word_count>109</wc:Word_count>
<wc:Classification_date>11-Nov-98 14:53:32</wc:Classification_date>
<wc:Last_modified>07-Aug-98 14:55:04</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://spsp.clarion.edu/mm/RDE3/start/">
                <wc:Accession_no>0</wc:Accession_no>
                <wc:Title>Research Design Explained 3rd ed</wc:Title>
                <wc:Abstract>Aids for teaching research methods in psychology</wc:Ab
                <wc:Keyword>
                        <rdf:Bag>
                        <rdf:li>computer</rdf:li>
                        <rdf:li>psychology</rdf:li>
                        <rdf:li>psychological</rdf:li>
                        <rdf:li>measure</rdf:li>
                        <rdf:li>experiment</rdf:li>
                        <rdf:li>experiments</rdf:li>
                        <rdf:li>research</rdf:li>
                        <rdf:li>learning</rdf:li>
                        <rdf:li>single</rdf:li>
                        <rdf:li>teaching</rdf:li>
                        <rdf:li>rights</rdf:li>
                        <rdf:li>writing</rdf:li>
                        <rdf:li>science</rdf:li>
                </rdf:Bag>
</wc:Keyword>
<wc:Classmark>
                <rdf:Seq>
                        <rdf:li>158     Applied psychology</rdf:li>
                        <rdf:li>150.724 Experimental research (Psychology)</
                </rdf:Seq>
                </wc:Classmark>
                <wc:Word_count>205</wc:Word_count>
                <wc:Classification_date>11-Nov-98 14:57:27</wc:Classification_date>
                <wc:Last_modified>31-Aug-98 12:53:37</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

Yahoo - Home : Reference : Libraries : Library and Information Science : Institutes

http://dir.yahoo.com/Reference/Libraries/Library_and_information_Science/Institutes

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://www.new-zealand.edu/infostudies/">
                <wc:Accession_no>0</wc:Accession_no>
                <wc:Title>Centre for Information Studies</wc:Title>
                <wc:Abstract>Courses on Information Literacy Staff Administration
                Current Courses Diplomas Certificates School based Courses Regional
                Courses Holiday Courses Courses for non teaching staff Newsletter We
                Tutorial</wc:Abstract>
                <wc:Keyword>
                        <rdf:Bag>
                                <rdf:li>librarianship</rdf:li>
                                <rdf:li>newsletter</rdf:li>
                                <rdf:li>education</rdf:li>
                                <rdf:li>school</rdf:li>
                                <rdf:li>administration</rdf:li>
                                <rdf:li>teacher</rdf:li>
                                <rdf:li>teaching</rdf:li>
                        </rdf:Bag>
                </wc:Keyword>
                <wc:Classmark>
                        <rdf:Seq>
                                <rdf:li>021     Relationships of libraries, archives
                                <rdf:li>370     Education</rdf:li>
                        </rdf:Seq>
                </wc:Classmark>
                <wc:Word_count>34</wc:Word_count>
                <wc:Classification_date>11-Nov-98 15:08:20</wc:Classification_date>
                <wc:Last_modified>15-Sep-98 22:25:51</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://www.mmu.ac.uk/h-ss/dic/">
                <wc:Accession_no>0</wc:Accession_no>
                <wc:Title>Department of Information and Communications MMU UK</wc:Ti
                <wc:Abstract>The Department of Information and Communications at
                the Manchester Metropolitan University UK Includes course research a
                contact details</wc:Abstract>
                <wc:Keyword>
                        <rdf:Bag>
                                <rdf:li>library</rdf:li>
```

```
                        <rdf:li>communications</rdf:li>
                        <rdf:li>school</rdf:li>
                        <rdf:li>university</rdf:li>
                        <rdf:li>science</rdf:li>
                        <rdf:li>management</rdf:li>
                </rdf:Bag>
        </wc:Keyword>
        <wc:Classmark>
                <rdf:Seq>
                        <rdf:li>380     Commerce, Communications, Transporta
                        <rdf:li>027     General libraries, archives, informa
                </rdf:Seq>
        </wc:Classmark>
        <wc:Word_count>7</wc:Word_count>
        <wc:Classification_date>11-Nov-98 15:16:18</wc:Classification_date>
                <wc:Last_modified>30-Jun-98 12:02:23</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

Yahoo - Home : Computers and Internet : Programming Languages

http://dir.yahoo.com/Computers_and_Internet/Programming_Languages/

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://www.ampl.com/cm/cs/what/ampl/">
        <wc:Accession_no>0</wc:Accession_no>
        <wc:Title>AMPL Modeling Language for Mathematical Programming</wc:Ti
        <wc:Abstract>FAQ BOOK SOLVERS PLATFORMS VENDORS CALENDAR MORE WHAT'S
        NEW EXTENSIONS CHANGE LOG REPORTS NETLIB EXAMPLES CONTENTS HOME AMPL
        Modeling Language for Mathematical Programming Try</wc:Abstract>
        <wc:Keyword>
                <rdf:Bag>
                        <rdf:li>computer</rdf:li>
                        <rdf:li>programming</rdf:li>
                        <rdf:li>modeling</rdf:li>
                        <rdf:li>communication</rdf:li>
                        <rdf:li>mathematical</rdf:li>
                        <rdf:li>model</rdf:li>
                        <rdf:li>models</rdf:li>
                        <rdf:li>control</rdf:li>
                        <rdf:li>linear</rdf:li>
                        <rdf:li>nonlinear</rdf:li>
                        <rdf:li>discrete</rdf:li>
                        <rdf:li>interface</rdf:li>
                        <rdf:li>web</rdf:li>
                        <rdf:li>com</rdf:li>
                        <rdf:li>language</rdf:li>
                </rdf:Bag>
        </wc:Keyword>
        <wc:Classmark>
                <rdf:Seq>
                        <rdf:li>004.6   Interfacing and communications (Comp
                        <rdf:li>005.1   Programming (Computer programming)</
```

```
                        </rdf:Seq>
                </wc:Classmark>
                <wc:Word_count>193</wc:Word_count>
                <wc:Classification_date>11-Nov-98 15:22:31</wc:Classification_date>
                <wc:Last_modified>08-Nov-98 23:39:21</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://quimby.fla.com/Activities/Programming/APRIL/
                <wc:Accession_no>0</wc:Accession_no>
                <wc:Title>APRIL</wc:Title>
                <wc:Abstract>Home page of the Network Agent Research Group within
                Fujitsu Laboratories of America Inc</wc:Abstract>
                <wc:Keyword>
                        <rdf:Bag>
                                <rdf:li>computing</rdf:li>
                                <rdf:li>programming</rdf:li>
                                <rdf:li>system</rdf:li>
                                <rdf:li>communication</rdf:li>
                                <rdf:li>model</rdf:li>
                                <rdf:li>internet</rdf:li>
                                <rdf:li>language</rdf:li>
                                <rdf:li>rights</rdf:li>
                                <rdf:li>interaction</rdf:li>
                                <rdf:li>america</rdf:li>
                        </rdf:Bag>
                </wc:Keyword>
                <wc:Classmark>
                        <rdf:Seq>
                                <rdf:li>003.5   Theory of communication and control
                                <rdf:li>005.1   Programming (Computer programming)</
                        </rdf:Seq>
                </wc:Classmark>
                <wc:Word_count>46</wc:Word_count>
                <wc:Classification_date>11-Nov-98 15:24:31</wc:Classification_date>
                <wc:Last_modified>Not known</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

Yahoo - Home : Society and Culture : Religion and Spirituality

http://dir.yahoo.com/Society_and_Culture/Religion_and_Spirituality/

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
```

```
<rdf:Description about ="http://www.theatlantic.com/election/connection/reli
        <wc:Accession_no>0</wc:Accession_no>
        <wc:Title>Political Issues Religion</wc:Title>
        <wc:Abstract>RELIGION Articles from The Atlantic Monthly 's archive
        and related links Welcome to the Next Church by Charles Trueheart 19
        Seamless multimedia worship round the</wc:Abstract>
        <wc:Keyword>
                <rdf:Bag>
                        <rdf:li>intellectual</rdf:li>
                        <rdf:li>multimedia</rdf:li>
                        <rdf:li>vision</rdf:li>
                        <rdf:li>archive</rdf:li>
                        <rdf:li>cultural</rdf:li>
                        <rdf:li>school</rdf:li>
                        <rdf:li>copyright</rdf:li>
                        <rdf:li>service</rdf:li>
                        <rdf:li>religious</rdf:li>
                        <rdf:li>university</rdf:li>
                        <rdf:li>public</rdf:li>
                        <rdf:li>family</rdf:li>
                        <rdf:li>church</rdf:li>
                        <rdf:li>worship</rdf:li>
                        <rdf:li>god</rdf:li>
                        <rdf:li>christian</rdf:li>
                        <rdf:li>christianity</rdf:li>
                        <rdf:li>spiritual</rdf:li>
                        <rdf:li>religion</rdf:li>
                        <rdf:li>America</rdf:li>
                        <rdf:li>politics</rdf:li>
                        <rdf:li>political</rdf:li>
                        <rdf:li>rights</rdf:li>
                        <rdf:li>communities</rdf:li>
                </rdf:Bag>
        </wc:Keyword>
        <wc:Classmark>
                <rdf:Seq>
                        <rdf:li>027     General libraries, archives, informa
                        <rdf:li>210     Philosophy and Theory of Religion</r
                </rdf:Seq>
        </wc:Classmark>
        <wc:Word_count>243</wc:Word_count>
        <wc:Classification_date>23-Nov-98 17:34:38</wc:Classification_date>
        <wc:Last_modified>Not known</wc:Last_modified>
</rdf:Description>
</rdf:RDF>
```

```
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
 xmlns:wc="http://scit.wlv.ac.uk/~ex1253/wc/schema/">
        <rdf:Description about ="http://www.inlink.com/~rife/religion.html">
                <wc:Accession_no>0</wc:Accession_no>
                <wc:Title>Dave's Controversial Religion Page</wc:Title>
                <wc:Abstract>Dave's Controversial Religion Page The Monroe
                Institute Spirit WWW site The Myth of the Historical Jesus Tibetan
```

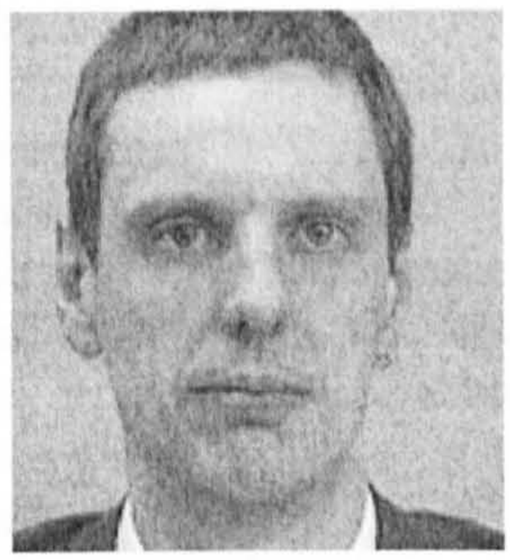
```
Book of the Dead Faqir Chand The Unknowing</wc:Abstract>
<wc:Keyword>
        <rdf:Bag>
                <rdf:li>jesus</rdf:li>
                <rdf:li>spirit</rdf:li>
                <rdf:li>religion</rdf:li>
        </rdf:Bag>
</wc:Keyword>
<wc:Classmark>
        <rdf:Seq>
                <rdf:li>210     Philosophy and Theory of Religion</r
                <rdf:li>290     Comparative Religion and Other Relig
        </rdf:Seq>
</wc:Classmark>
                <wc:Word_count>76</wc:Word_count>
                <wc:Classification_date>23-Nov-98 17:37:05</wc:Classification_date>
                <wc:Last_modified>Not known</wc:Last_modified>
        </rdf:Description>
</rdf:RDF>
```

Vitae

Charlotte Jenkins is a Research Student at the University of Wolverhampton, UK. Her research is concerned with tools for information resource discovery on the Web and in particular automatic classification. Charlotte graduated from Oxford Brookes University in 1995 with a B.Sc. Joint Honours in Computing and English studies

Mike Jackson is Professor of Data Engineering at the University of Wolverhampton, UK. He is a Fellow of the British Computer Society. He obtained his first degree in Computer Science at Sheffield City Polytechnic and his Masters at Manchester University. Mike has served on the organising and programme committees of numerous database conferences including BNCOD, IDEAS, EDBT, ICDE, ER and VLDB.

Peter Burden graduated with a BA in Mathematics from the University of Cambridge in 1964. He is currently employed in the School of Computing and Information Technology at the University of Wolverhampton where he teaches systems and network programming and is responsible for the School's Unix based systems. His research interests include Internet resource discovery and cataloguing.



Jon Wallis works as an IT consultant in the pharmaceutical industry, specialising in laboratory automation and robotics. Prior to this, Jon was a senior lecturer at the University of Wolverhampton. His teaching was mainly in the area of computer networks and communication systems, with research interests in web search engines and the information management issues of corporate web sites.

The Wolverhampton Web Library (WWLib) and Automatic Classification

Charlotte Jenkins, Mike Jackson,
Peter Burden, Jon Wallis
School of Computing & IT
University of Wolverhampton
Wulfruna Street, Wolverhampton
WV1 1SB, UK

The Wolverhampton Web Library (WWLib) is a World Wide Web search engine that classifies UK Web pages according to Dewey Decimal Classification (DDC)[1]. The original version was developed in 1995 as a result of poor response times, US bias and information overload□ from the big US search engines. The decision to use DDC evolved from the notion that library science - that has been responsible for organising vast amounts of information for decades - has a lot to offer the comparatively chaotic task of information resource discovery on the Web.

Tools for locating information on the Web have evolved from manually maintained classified directories like GENVL[2], Galaxy and Yahoo, into fully automated search engines like Alta Vista, Lycos, Excite, Infoseek, HotBot etc [3]. Classified directories provide access to manually classified documents that are clustered according to a pre-prescribed classification scheme. Automated search engines use a robot to retrieve documents from the Web, which are then automatically analysed and used to generate huge unclassified indexes. The advantage of automated tools is that they provide much more comprehensive Web coverage and are generally more up-to-date due to the continuous activity of their robots. The lack of classification and human intervention, however, appears to result in a tendency to overload users with irrelevant, poor quality results. In contrast, Yahoo has maintained its popularity as a manually maintained classified directory, because it provides very accurate high quality information and it is intuitive to use. Classified tools usually enable users to browse a classification hierarchy where it is possible to focus their query on certain subject areas. Results are then restricted to those subject areas making the occurrence of irrelevant results and information overload very uncommon.

Many automated search engines have turned to traditional Information Retrieval (IR) research in an attempt to improve the accuracy of their results. Few, however, have considered the work on automatic classification that Good[4], Fairthorne[5] and Salton[6] were discussing in the early 1960s. Automatic classification[7] has the potential to combine the advantages of classified directories with the advantages of automated search engines and result in an accurate, intuitive, comprehensive, classified search engine.

The original version of WWLib relied to a large degree on manual maintenance and as such can best be described as a classified directory that organised resources according to DDC. A new fully automated version is being designed and developed, WWLib TNG (The Next Generation), which will support a robot, automatic indexing and an automatic classifier. The use of DDC is appropriate for a number of reasons: It is a universal classification scheme covering all subject areas and geographically global information. Users who are accustomed to using a library will find the classification system familiar and DDC has multilingual scope, which will become increasingly important as the volume of information in other languages grows on the Web. The hierarchical nature of DDC makes it easier to move from rough classifications to increasingly more accurate ones.

This paper will describe the WWLib TNG architecture and approach. The design and implementation of the automatic classifier, in particular, will be discussed in detail. The classifier has been developed in Java and combines IR research with library science by automatically classifying Web documents according to DDC.

1. L. Mai Chan, J. P. Comaromi, J. S. Mitchell, M. P. Satija, Dewey Decimal Classification: A Practical Guide, Forest Press, ISBN 0-910608-55-5, 1996

2. O. A. McBryan, GENVL and WWWW: Tools for Taming the Web http://www.cs.colorado.edu/home/mcbryan/mypapers/www94.ps From proceedings of the First International World Wide Web Conference, ed. O. Nierstrasz, CERN, Geneva, May 1994

3. C. Jenkins, M. Jackson, P. Burden, J. Wallis, Searching the World Wide Web An Evaluation of available Tools and Methodologies, (accepted for publication in the journal Information and Software Technology)

4. J. Good, Speculations Concerning Information Retrieval, Research Report PC-78, IBM Research Center, 1958

5. R. A. Fairthorne, The mathematics of classification: Towards Information Retrieval, Butterworths, 1961

6. G. Salton, Automatic Information Organization and Retrieval, McGraw-Hill, New York, 1968

7. R. C. J. van Rijsbergen, Information Retrieval: Second Edition, Chapter 3, http://www.dcs.glasgow.ac.uk/Keith/Chapter.3/Ch.3.html, Butterworths, ISBN 0-408

Automatic classification of Web resources using Java and Dewey Decimal Classification

Charlotte Jenkins, Mike Jackson, Peter Burden, and Jon Wallis

School of Computing & IT, University of Wolverhampton
Wulfruna Street, Wolverhampton, WV1 1SB, UK

Abstract

The Wolverhampton Web Library (WWLib) is a World Wide Web search engine that provides access to UK based information. The experimental version, developed in 1995, was a success but highlighted the need for a much higher degree of automation. An interesting feature of the experimental WWLib was that it organised information according to Dewey Decimal Classification (DDC)[1]. This paper discusses the advantages of classification and describes the automatic classifier that is being developed in Java as part of the new, fully automated WWLib.

Keywords

Search; Retrieval; Classification

1. Introduction

The advantages of document clustering and classification over keyword based indices have been debated in Information Retrieval (IR) research for quite some time. Documents that share the same frequently occurring keywords and concepts are usually relevant to the same queries. Clustering such documents together enables them to be retrieved together more easily and helps to avoid the retrieval of irrelevant unrelated information. Another advantage is that classification usually enables the ability to browse through a hierarchy of logically organised information which is often considered a more intuitive process than constructing a query string. Keyword indices are however comparatively simple to construct automatically. Consequently, classification is usually associated with human defined metadata or catalogue entries.

The evolution of automated World Wide Web search engines from manually maintained classified lists and directories has further demonstrated the strengths and weaknesses of these two approaches. The tendency of automated search engines to inundate users with irrelevant results has prompted reconsideration of the merits of classification. The combination of automation and classification has the potential to provide an accurate, intuitive, comprehensive classified search engine. This is the aim of WWLib.

2. WWLib

The original version of WWLib relied to a large degree on manual maintenance and as such can best be described as a classified directory that was organised according to DDC. The use of DDC to organise WWLib evolved from the notion that Library Science has a lot to offer the chaotic task of information resource discovery on the Web. The classified nature of WWLib was beneficial in that it clustered documents according to subject matter and enabled users to browse through documents that shared the same DDC classmark as those that appeared in the results of a query.

It was soon evident, however, that WWLib required a much higher degree of automation. A robot for resource discovery and an automatic indexer were required but the automated WWLib would preserve its classified nature by employing an automatic classifier. An outline design of the new automated WWLib, shown in Fig. 1, identifies the automated components and their responsibilities:

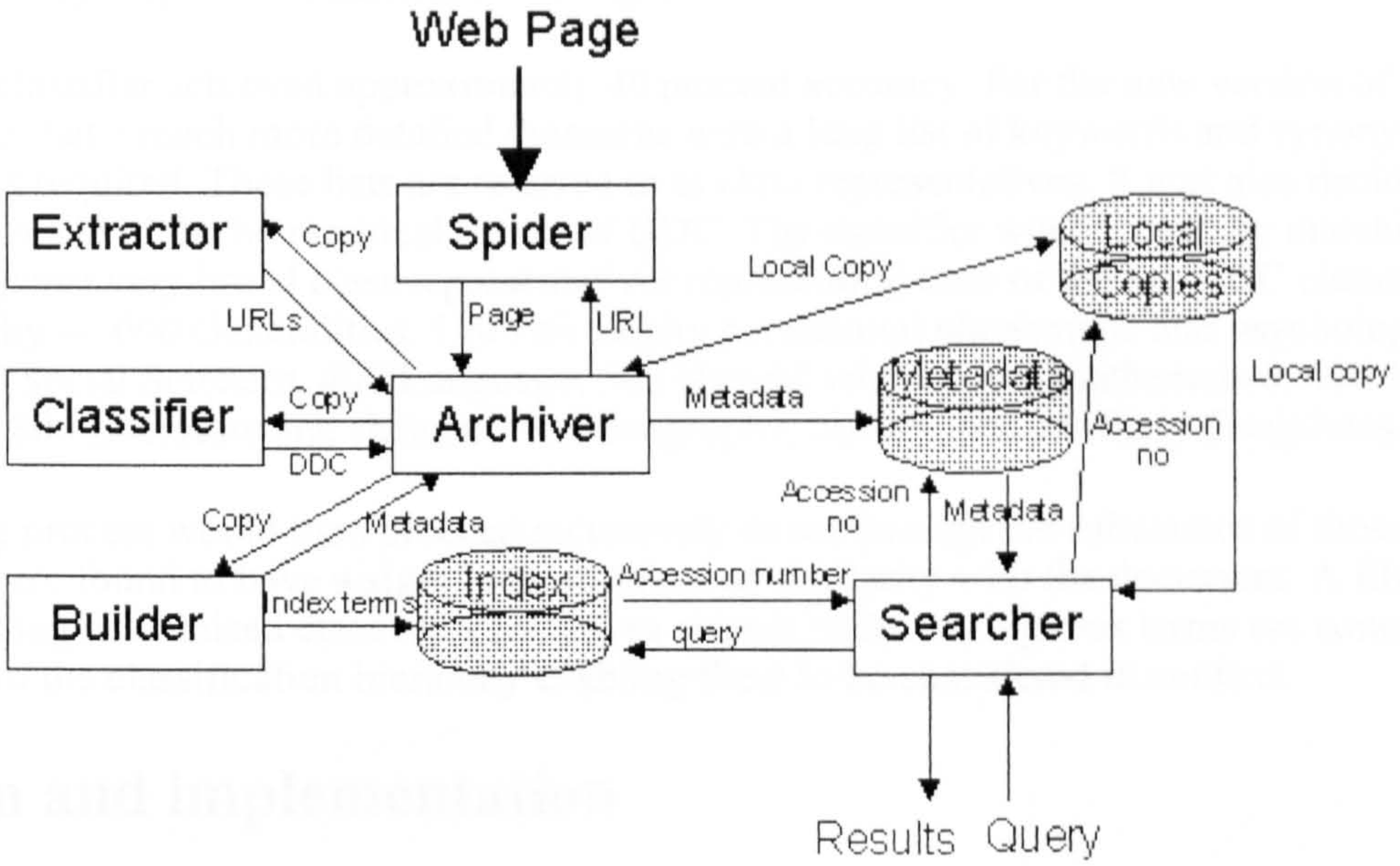


Fig. 1. Overview of the new WWLib architecture.

There are six automated components:

1. A **Spider** that automatically retrieves documents from the Web;
2. An **Archiver** that receives Web pages from the spider, stores a local copy, assigns to it a unique accession number and generates a new metadata template. It also distributes local copies to the Extractor, Classifier and Builder and adds subsequent metadata generated by the Classifier and the Builder to the assigned metadata template;
3. An **Extractor** that analyses pages, provided by the archiver for embedded hyperlinks to other documents. If found, URLs are passed to the archiver where they are evaluated to check that they are pointing to locations in the UK, before being passed to the Spider;
4. A **Classifier** that analyses pages provided by the archiver and generates DDC classmarks;
5. A **Builder** that analyses pages provided by the archiver and outputs metadata which is stored by the archiver in the document's metadata template and is also used to build the index database that will be used to quickly associate keywords with document accession numbers;
6. A **Searcher** that accepts query strings from the user, uses them to interrogate the index database built by the builder, uses the resulting accession numbers to retrieve the appropriate metadata templates and local document copies and then uses all this information to generate detailed results, ranked according to relevance to the original query.

One of the reasons for deciding on such a componentised architecture was to allow for components to be distributed over a network if necessary.

3. The classifier

Many automated search engines have deployed traditional IR indexing strategies and retrieval

mechanisms but very few have experimented with automatic classification. Previous experimentation with automatic classification was carried out during the development of the original WWlib. This original classifier[2] compared text in each document with entries in a *DDC thesaurus* file. The thesaurus entries consisted of the DDC classmark and accompanying header text eg:
641.568 Cooking for special occasions Including Christmas

The original classifier achieved approximately 40 percent accuracy. For the new version of the classifier, it was decided that a much more detailed thesaurus with a long list of keywords and synonyms for each classmark was required. These lists are referred to as class representatives. It was also decided that more use would be made of the hierarchical nature of DDC. The classifier would begin by matching documents against very broad class representatives representing each of the ten DDC classes at the top of the hierarchy — 000 Generalities, 100 Philosophy paranormal phenomena and psychology, 200 Religion, 300 Social Sciences, 400 Language, 500 Natural sciences and mathematics, 600 Technology, 700 The arts, 800 Literature and rhetoric, 900 Geography, history, and auxiliary disciplines.

The matching process would then proceed recursively down through the subclasses of those DDC classes that were found to have a significant measure of similarity with the document. A filtering effect is achieved using customised class representatives at each node. Ambiguous terms are concealed within lower nodes of the classification hierarchy enabling them to be considered in context.

4. Design and implementation

The classifier has two main processes; firstly the document is indexed; secondly the document is classified. The indexing process results in the formation of a document object. The document object comprises a number of keyword objects, each one representing a word found within the document. Keywords have a weight — assigned according to where the word was found — and a position associated with them.

The classification process uses a classify object which takes the newly formed document object and compares it with a number of DDC objects. DDC objects inherit their structure and behaviour from an abstract class *Dewey*. They too are made up of a series of weighted keyword objects that together make up the class representative. Each DDC object has a classmark object specifying its dewey decimal classmark, and can have up to ten subclasses which are in themselves DDC objects representing the next layer of the hierarchy. The classify object begins by comparing the document object with the ten DDC objects representing the top of the DDC hierarchy. If the document matches significantly with a DDC object, instances of that DDC object's subclasses are created and the document is compared with those. This process continues recursively down the hierarchy until a significant match is found with a leaf node (a DDC object with no subclasses). In this event the classmark object belonging to the DDC objec is copied into the document object. Measures of similarity are calculated using the Dice Coefficient[3]. The indexing and classification processes are co-ordinated by the Ace (Automatic classification engine) object.

The classifier has been implemented in Java. *This has* enabled easy networking, multithreading and memory management.

5. Conclusion

The new classifier is in the early stages of evaluation. It appears, however, that use of a hierarchical classifier results in context sensitive classifications. The use of manually defined class representatives,

that perform context sensitive filtering, encourage accuracy. To increase the accuracy of the classifier further a more comprehensive set of DDC class representatives are required. When sufficient DDC classes have been defined, formal testing will be required to prove that the new classifier is achieving a higher rate of accurate classifications than the original one. There is a working paper that describes the design and implementation of the classifier in more detail.

References

1. L. Mai Chan, J.P. Comaromi, J.S. Mitchell and M. P. Satija, *Dewey Decimal Classification: A Practical Guide*, Forest Press, ISBN 0-910608-55-5, 1996.
2. J. Wallis, P. Burden, Towards a classification-based approach to resource discovery on the web, University of Wolverhampton, 1995, http://www.scit.wlv.ac.uk/wwlib/position.html
3. R.C.J. van Rijsbergen, *Information Retrieval: Second Edition* (Chapter 3, http://www.dcs.glasgow.ac.uk/Keith/Chapter.3/Ch.3.html), Butterworths, London, ISBN 0-408-10775-8, 1981.

URL Section

- The Wolverhampton Web Library (WWLib)
  http://www.scit.wlv.ac.uk/wwlib/
- Automatic Classification of Web resources Using Java and Dewey Decimal Classification
  Working paper
  http://www.scit.wlv.ac.uk/~ex1253/classifier/

Searching the World Wide Web

An Evaluation of available Tools and Methodologies

Charlotte Jenkins, Mike Jackson,

Peter Burden, Jon Wallis

University of Wolverhampton

School of Computing & IT
University of Wolverhampton
Wulfruna Street
Wolverhampton
WV1 1SB
Tel. 01902 321405

Abstract

Search Engines and Classified Directories have become essential tools for locating information on the World Wide Web. A consequence of increasing demand, as the volume of information on the Web has expanded, has been a vast growth in the number of tools available. Each one claims to be more comprehensive, more accurate and more intuitive to use than the last. This paper attempts to organise the available tools into a number of categories, according to their information acquisition and retrieval methods, with the intention of exposing the strengths and weaknesses of the various approaches. The importance and implications of Information Retrieval (IR) techniques are discussed. Description of the evolution of automated tools enables an insight into the aims of recent and future implementations.

Keywords: *Web, Search, Retrieval*

1 Introduction

Information resource discovery on the World Wide Web is a significant task; the growth of the information available and the increasing number of users requiring simultaneous access to it, are two issues that continuously add to its complexity. This paper is concerned with the former issue - tools for retrieving information from a resource that is continuously growing and changing.

Prior to the Web, the most popular method for retrieving information from the Internet was Gopher [1]. Plain text files, image and sound files were organised by category into hierarchical structures on Gopher servers which were then accessed using a Gopher client. Hierarchical menus and sub-menus led the user to the required information in an uninspiring but organised, logical fashion. Information was grouped by category and the items on each server were registered with the *Mother of all Gophers* [2] in Minnesota. A search mechanism known as *Veronica* [2] could be used to interrogate the information held by the Mother Gopher and find the location of required information.

The notion of maintaining a central resource, the *Mother of all Gophers*, was appropriate before the Web when there was much less information available. Publishing information on Gopher required prior knowledge of certain configuration details which, along with the fact that it was visually unimpressive, prevented it from ever appealing to the masses. A comparatively small collection of unimaginatively formatted but reasonably well organised information has given way to a mass of hyperlinked information with no central resource and no simple method for locating

anything. Classified directories with hierarchical classification schemes and fully automated search engines have evolved in an attempt to solve this problem.

2 Classified Directories

An early attempt was made to emulate the logical structure of Gopher on the Web by GENVL - Generate Virtual Library [3]. GENVL was named the *Mother of all Bulletin Boards* because, like Gopher, it built a hierarchy of user-supplied virtual sub-libraries. The concept was the same as Gopher but GENVL did not have the monopoly that the *Mother of all Gophers* had. Pages that were not registered with GENVL were often equally as popular as pages that were. In fact, there were no laws of convention associated with the Web at all and the ease with which one Web page could provide hyperlinks to many others eliminated dependency on a single central register.

The hierarchical, classified nature of GENVL was exploited to the advantage of a number of later tools, most notably Yahoo (see table 1). Yahoo is still respected as one of the best manually maintained classified tools for locating information. It has its own proprietary classification scheme under which Web resources are grouped. Adding new pages to Yahoo is comparatively simple; an HTML form provides the means by which users simply select a category from Yahoo's pre-prescribed classification scheme. Data taken from users about each document is used to build a database and users then wishing to locate required information are given the choice to either browse the classification hierarchy or query the database with a query string. In either case this classified approach has the advantage of displaying clusters of documents that have been manually classified into the same category.

Early classified directories attempted to solve the problem of resource discovery by employing human "scouts" who spent their time browsing for new sites. The popularity of Yahoo became such that scouts were given the job of rating and reviewing user-supplied pages rather than looking for new material. Although more recent automated tools with their Web roaming robots and automatic indexing (see next section) such as Lycos, Alta Vista and Excite looked as though they might leave the manually maintained directories behind, the merits of classification and human defined metadata have since re-emerged as very important issues and Yahoo has maintained its popularity.

Recent arguments for manually maintained classified directories centre around the notion that they provide quality rather than quantity. An appropriate category for each new resource is chosen by the user (often the author of the page) manually and keywords for indexing purposes are also entered manually. This human interaction combined with a well structured classification hierarchy usually ensures that users seeking information are not inundated with irrelevant, misleading suggestions, as they often are with automated, unclassified tools. The ability to browse the classification hierarchy is considered more intuitive for novice users[4]. Advanced users, who might find browsing frustrating, have the option to enter a query string.

The most damning criticism of Yahoo style classified directories is that, due to their manual data acquisition and maintenance, they suffer from poor Web coverage and out of date information[4]. Queries are matched only against the keywords, descriptions and titles entered by the user and not the full text of the document which means relevant documents could be missed unless they happen to appear in the same category as those that are found. Classified tools that offer rating and reviewing as part of the service have been criticised for constraining query results and

inducing biased information [4], suggesting that too much human intervention is undesirable. In answer to criticisms such as these automated search engines were introduced.

Table 1 shows a comparison of a number of classified directories.

3 Automated Search Engines

Experience with GENVL showed that it was insufficient to rely entirely on user submission for resource discovery. In answer to this the World Wide Web Worm was developed - the first automated Search Engine. It worked by using what is now commonly known as a *robot* or *spider* - in other words a mechanism for retrieving documents from the Web and analysing them for embedded URLs (hyperlinks within the document that lead to other documents). When embedded URLs were found, those documents would be retrieved and analysed for further URLs and so on until the whole Web had been retrieved. A database was maintained that kept a record of each URL and where it was found. The World Wide Web Worm's user interface provided the means by which a query string could be entered into the input field of an HTML form which, when submitted, would be used to query the database. The results from such a query comprised a list of sites whose URL, title or heading fields were found to contain some or all of the terms in the query string. Additionally, each result was accompanied by the URL of the page in which the document was cited providing a citation index.

The World Wide Web Worm won *The Best of the Web* award in 1994. The concept was taken on and improved by other mechanisms such as Lycos and Infoseek who developed more rigorous robots and more comprehensive IR text analysis techniques that maintained more comprehensive databases. Table 2 lists the major search engines chronologically with dates.

Indexing the entire Web, even using a robot, is an impossible task. The Web is always changing, every minute new pages are added while old ones are changed or deleted. Robots are often employed in revisiting known resources to detect changes, as well as discovering new ones, in an attempt to cope with this transient behaviour. "Dead links" - links to pages that no longer exist - are a problem for search engines and classified directories but they are noticeably more common in the results of classified directories because of their inability to perform automatic database updates. Some search engines maintain a record of how often sites update their information and consequently revisit sites that are constantly changing more regularly.

There are certain codes of conduct governing the behaviour of robots. A text file called robots.txt placed in the root directory of any Web server can be used to specify areas of the server that may not be accessed by a robot. This is known as robot exclusion. Martijn Koster of Nexor, also the author of the Aliweb (http://www.nexor.co.uk/public/aliweb/search/doc/form.html) search engine, was involved in the specification of a standard for robot exclusion which has now been incorporated into the HTML 4.0 standard. He also maintains The Web Robots Pages [5] at WebCrawler where the activity of all known robots is monitored. New robots can be registered on this site and there is a wealth of advice for robot writers. In addition to robot exclusion there are other conventions that robots should observe, such as allowing a time delay between requests to avoid "rapid-fire" on a server. Server administrators can report robots that do not observe the agreed protocol to The Web Robots Pages and also on a Robot Alert mailing list that can be found at http://www.zyzzyva.com/robots/alert/.

After pages have been discovered by a robot they must then be indexed. Classified directories obtain keywords for indexing purposes from the user when the document is submitted, search

engines must index documents automatically. Some search engines such as Alta Vista boast full text indexing which means virtually every word in every document is matched against user queries. Although this means that relevant documents are rarely overlooked, it also means that irrelevant ones, that happen to contain certain relevant words out of context, can be retrieved if terms are not weighted according to significance (see later section on indexing). Alta Vista was very well received initially because of its very powerful and effective robot. Unfortunately, a good robot combined with full text indexing and a comparatively poor retrieval mechanism leads to high recall but low precision, in other words information overload. Other search engines, such as Lycos, use tried and tested IR indexing algorithms to deduce keywords that are particularly relevant to the subject of a document, resulting in more accurate indexing. Indexing strategies are discussed in more detail in the later section on IR.

Once documents have been indexed, information needs to be stored about each resource. This is known as metadata[6]. Typical metadata will include the title, URL, IP address, summary or description, keywords or index terms, file size, last modified date, the date the resource was first discovered, the date it was last checked for validation and so on. Although there are metadata standards (see later section on the future), most search engines define their own proprietary templates. This information then needs to be stored in such a way that the retrieval mechanism has fast and easy access to the index terms.

The retrieval mechanism facilitates the identification and retrieval of documents relevant to user queries. Various approaches to the retrieval and ranking of results are utilised. Often the retrieval mechanism is heavily dependent on the indexing strategy (both these issues are discussed in the later section on the Role of IR). Depending on the retrieval mechanism, advanced search options including boolean syntax and/or phrase matching or natural language processing may be available to the user submitting a query.

The user interface plays an important role in obtaining a well focused query from the user and it is also important in the presentation of query results. The items resulting from a query are usually organised into some kind of rank order by the retrieval mechanism and are then presented to the user, a number at a time, with the most relevant appearing first. Metadata about each retrieved document is presented with the title appearing as a hyperlink to the document itself. The amount of metadata displayed varies from one search engine to the next depending on the information that is originally stored when indexing. It is important that results are clear and concise with well described items.

In summary, automated search engines generally comprise the following components:

- A robot that continually retrieves documents and analyses them for hyperlinks to other documents in an attempt to provide comprehensive Web coverage;
- An indexer that uses an IR indexing strategy to extract accurate index terms from the document;
- A database where metadata describing each resource is stored;
- A retrieval mechanism that takes user queries and quickly retrieves and ranks relevant documents from the database;
- A good user interface that encourages the user to input a coherent, well focused query and subsequently presents a clear set of results.

A number of automated search engines also offer a browsable classified directory, the entries of which are usually a subset of those found in the search engine database that are considered worthy of notice by staff who manually maintain the directory.

The main criticism of automated tools is that they tend to overload users with irrelevant, misleading results. Complex boolean syntax is often required to focus queries appropriately which can be very confusing for novice users. Due to the lack of human intervention, results often contain links to very poor quality information and potentially useful information can be very badly indexed and described. The lack of classification can lead to documents that happen to share the same relevant words but not necessarily shared relevant context being displayed next to each other in the results.

Table 3 shows a list of automated search engines with a comparison of available features.

4 Other approaches

4.1 Meta Search Engines

Meta Search Engines provide the interface for querying the databases of a number of search engines and classified directories from the same page. The service provided by these tools varies considerably. Some provide a series of direct links to a large selection of search engines, others provide one input field and query a series of databases more transparently. Querying each search engine individually usually results in the user interacting with each search engine directly. Those that take one query string and submit it to several tools often post process the results by collating *and ranking them. This is perhaps a more useful service but the overhead involved is obviously* considerable. The degree to *which queries are pre-processed - translated into the correct* syntax for each search engine - is not clear; in most circumstances complex boolean queries are *not* advisable via meta search engines. Some users take advantage of the larger bandwidth of a local meta search engine to access remote resources (in the US). The number of databases queried varies from one meta engine to the next with some just querying the most prevalent - Alta Vista, Lycos, Infoseek, Excite - and others querying a longer and more varied list of search engines.

Table 4 lists some of the available meta search engines.

4.2 Geographically Specific Resources

Most major search engines and classified directories such as Alta Vista, Excite, Lycos, HotBot Infoseek, Yahoo, Galaxy, Magellan... and so on, are situated in the USA. Internet users in other parts of the world often have problems with this due to poor response times, particularly in the afternoon (GMT) when the US are awake and transatlantic traffic becomes exceptionally congested. A tendency to provide US biased information can also be a problem. US bias has come about, not only because most of the major search engines are located in the US, but also because the US are currently making more extensive use of the Web than most other places. US information tends to drown out most other information, simply because there is more of it.

A growing number of local search engines have emerged in the UK, Europe and other parts of the world that provide information on the local domain. Some people believe it is more beneficial to mirror the big US engines locally than to keep reinventing the wheel by developing more and more new search engines. It may be that the well established search engines have better resources and can therefore afford to provide a better service with bigger, faster machines and faster, higher bandwidth connections. The disadvantage of mirroring, however, is that

although the response time problem is solved, the database in most cases remains the same - US biased results with US biased reviews.

Table 5 shows some of the many geographically specific resources.

4.3 Subject Specific Resources

Even the most comprehensive automated search engines cover just a small proportion of the total amount of information available. Covering the Web in its entirety is an impossible task due to its constantly changing nature. Although automated tools are more comprehensive than manual ones it seems the less human intervention the less accurate the results. A proposed method for improving coverage, at the same time as solving other problems such as information overload, poor quality information and irrelevant query results, is to provide a series of directories that are each dedicated to a specific subject area. Each directory is maintained by experts in the particular subject area who provide site reviews and ratings and ensure that accurate, high quality information is maintained in a well structured hierarchy.

This concept has been taken up quite seriously by a number of specialist groups. The Resource Organisation And Discovery System (ROADS) [7] has encouraged the development of a number of high quality information gateways. These are discussed in more detail in the later section on the future.

Table 6 lists some subject specific gateways.

5 The Role of IR

Evaluation of tools for information retrieval is usually based on two measures - recall and precision. Recall refers to the percentage of all relevant documents that are retrieved from a database and precision refers to the percentage of the documents retrieved that are relevant. For example, if documents on Medieval English Literature were sought from a database that contained 80 documents relevant to this query, 20 of which were retrieved along with 30 irrelevant ones - 50 documents being returned in total - recall and precision would be calculated as follows [8]:

$$\text{Recall} = \frac{\text{Number of items retrieved that are relevant}}{\text{Total number of relevant documents in the database}} = \frac{20}{80} = 0.25$$

$$\text{Precision} = \frac{\text{Number of items retrieved that are relevant}}{\text{Total number of documents retrieved}} = \frac{20}{50} = 0.4$$

There are two areas of search engine functionality that prescribe the degree of recall and precision; the indexing strategy and the retrieval mechanism.

5.1 Indexing

IR indexing strategies have evolved from the manual task of library cataloguing where librarians would manually specify a number of keywords to identify each item (book, journal, etc.). The performance of a search engine, in terms of recall and precision, relies heavily on its indexing strategy; what information is extracted from each document and how accessible that data then is are crucial issues.

There are generally two types of automatically generated index; weighted and unweighted[9]. In an unweighted index each term is stored with a value describing its location and little or no further information. These indexes best support boolean searches where a document is either relevant or it is not. No indication as to the degree of relevance can be easily obtained from this kind of index.

With a corpus the size of the Web it is obviously advantageous to organise results from a query into a ranked list with documents that are likely to be most relevant at the top. In a weighted index terms are assigned a weight according to their frequency within the document. Luhn, Brookstein, Klein and Raita's theories[9] all support the notion that the significance of a word, in terms of its power to reveal concepts within a document, is directly proportional to the frequency with which it occurs within the document. Weight values assigned to index terms are commonly normalised to a figure between zero and one, one indicating the highest significance. The number of occurrences of the term in the database as a whole is often used to avoid common 'stop' words being assigned a significant weight value. This weighting enables the retrieval mechanism to score and rank documents according to their relevance to the user query. Often query terms are themselves weighted to identify the most important words in terms of their power to retrieve relevant documents. This is done by assigning weights according to word frequencies within the database.

The Vector Space Model[9, 10] is a common IR approach to weighted indexing and subsequent retrieval. Documents are represented as vectors, each of which have a vector position for every known term (word) in the database. The indexing mechanism assigns a weight to the position of each found term depending on its frequency. Terms that are not found have a value of zero. Queries are then also translated into vectors so that a measure of similarity between the query vector and the document vector can be obtained. A variant of this approach is known to be used by the Excite search engine as part of its Intelligent Concept Extraction (ICE) process[10].

Another common IR approach is based on a probabilistic model, the most common of which is known as the Bayesian Model[9, 11], whereby the probability of a document containing a particular concept is calculated on the basis that it contains certain words.

Some search engines claim to use natural language processing. This is where constructs within the language are identified; semantic information is combined with statistical information to identify phrases and word patterns. The frequent co-occurrence of terms across a range of documents is also used to identify phrases and concepts.

It is important that once the indexing terms have been ascertained, they are stored in a manner that enables fast access by the retrieval mechanism. A common method of quickly associating query terms with document accession numbers is to use an inverted file index. This is where every possible term has an entry in an index file with a list of associated document accession numbers. These accession numbers can then be used to look up further metadata and often a local copy of the full text of the document.

5.2 *Retrieval*

Retrieval algorithms used by the searching component of search engines generally fall into one of three areas; boolean, probabilistic or natural language processing.

Boolean Searches

Many search engines encourage the use of boolean syntax within user queries. A user wishing to locate documents about 'Equine Anatomy' might find that merely typing the two search terms into the query input box of some search engines leads to results referencing hundreds of pages about horses, but not anatomy, and/or hundreds of pages about anatomy, but not horses, drowning out the few relevant pages that are actually about equine anatomy. The query string 'equine AND anatomy' (capitalisation of operators seems to be common but not all search engines use this syntax) would probably be far more successful as only those documents containing both terms would be retrieved. The boolean operators AND, OR and NOT are implemented using intersection, union and difference procedures from set theory.

Boolean logic provides a means for focusing queries well and can help to improve recall and/or precision. Sometimes it is possible to include parentheses to dictate the order of operators. For example, if a user wanted to retrieve documents about reptiles and/or mammals but not humans they could use the query 'reptile OR (mammal NOT human)'.

Fuzzy Boolean

When a query string contains more than one term, in the absence of any boolean operators, Fuzzy Boolean[10] is often used. Documents are ranked according to the number of terms matched. This tends to improve precision at the top of the list.

The term *Fuzzy Searching*[9] is used to describe a mechanism that is often used as a result of very poor recall. Terms that have similar spelling to the query terms are sought in the assumption that the query terms have been incorrectly spelt.

Proximity searching and phrase matching

Some tools provide a mechanism for specifying that the search terms entered must appear adjacent to each other. This is usually indicated by the adjacency operator, ADJ. Proximity searches may also be encountered that specify that terms must occur 'near' each other. It is generally considered that two terms occurring near to each other give better indication of concept[9]. For example, a document containing the term 'historical' close to the term 'architecture' is more likely to contain information about historical buildings than a document that has these terms several paragraphs apart.

Proximity searches are usually based on the proximity of just two terms. If a user wanted to search for a whole phrase, it is often possible to enclose a phrase such as 'Child Development Psychology' in inverted commas, only those documents containing the exact phrase should then be retrieved.

Thesaurus searches and query expansion

One method of improving recall is to retrieve documents that, not only contain the query terms, but synonyms of those terms also. Electronic thesauri are available to enable this process. The problem with this approach is that often the focus of queries can be badly skewed by unsuitable synonyms resulting in improved recall but disastrously low precision. To avoid this negative result search engines supporting this feature often present the user with a list of synonyms relating to their original terms so that they can select relevant ones. Alta Vista's *Live Topics* is an example of this approach.

Statistical thesauri provide an alternative method. Instead of looking up semantic synonyms, terms that have a statistically high coincidence with the user's query terms within documents are added to the query. This approach is known as automatic query expansion. Often the original query is processed to reveal which terms are most likely to focus the query - those that are less common within the database - and the query is then expanded with statistical synonyms of those terms. The Muscat[12] search engine, EuroFerret (see table 5) uses probabilistic retrieval in conjunction with 'Relevance Feedback'. This enables the user to indicate which results are most relevant to their query and similar documents with a high coincidence of significant terms are then sought.

Stemming and term masking

The retrieval mechanism may also improve recall by carrying out suffix and gerund stripping or 'stemming' on the query string. This means that any terms ending in "s", "ed", "ing", "ology", "ologist", "ological" etc. will be stripped so that, for example, a search for "Psychological Conferences" will find a document containing the words "Psychology Conference" highly relevant.

Stemming is often used to improve recall but it can have a negative effect on precision. The Porter stemming algorithm[13] identifies words with certain suffixes and replaces them with stemmed versions. This can result in decreased precision, as Kowalski[9] points out 'memorial' and 'memorise' have very different meanings but would both be reduced to 'memory' by the Porter algorithm. An alternative method is to use a dictionary based approach such as Kstem[9] where more accurate stems are obtained by replacing the word with the most appropriate stem obtained from a dictionary. Frakes'[14] evaluation of stemming experiments confirmed that stemming algorithms only have a positive effect on recall, not on precision.

A different approach altogether is to use term masking[9] in the query. The endings of words are masked and any combination of characters after the unmasked characters can be accepted as a match. For example the masked term psycho* could be matched against the terms psycho, psychology, psychologist, psychological and so on.

6 The Future

Recent developments in information resource discovery on the Web, have aimed to combine the comprehensive Web coverage of automated search engines with the accuracy and intuitiveness of manually maintained classified directories. Some recent projects with this main aim are detailed below.

6.1 Subject Specific Databases

One method of providing high quality information and comprehensive coverage simultaneously is to maintain separate subject specific databases that can be queried seamlessly from the same interface. ROADS and The Search Broker are two examples of this approach.

ROADS

ROADS - Resource Organisation And Discovery in Subject-based services [7] - is a UK based project of eLib, the Electronic Libraries association. The system provides a mechanism for well informed experts in a particular field to manually maintain a database of quality information in their given field. The ROADS software provides experts with an intuitive interface for maintaining a well organised hierarchy of information.

Each ROADS database is made up of a series of metadata templates, each one representing a different resource. The Internet Anonymous FTP Archive Template (IAFA Template[15]) is used as the metadata standard across all ROADS databases. The use of a standard template for database records means that all ROADS databases are compatible with each other. Eventually, a top level gateway will provide seamless access to all the sub-gateways, each maintained by a different group of specialists. This should provide comprehensive coverage of accurate, high quality information.

The Electronic Libraries association (eLib) and the On-line Computer Library Centre (OCLC) in Dublin, Ohio, have been involved in developing new metadata standards to encourage interoperability among such subject based resources. The Dublin Core[16] metadata standard is becoming the most common template.

The Search Broker

Developed by the University of Arizona, this system[17] encompasses over 300 *search servers* - distributed databases that each specialise in a particular subject area. Search servers can be generated using their own Glimpse[18] or GlimpseHTTP software which is also the indexing mechanism behind Harvest[19].

Search Broker queries require the specification of a topic or subject area as well as the usual query terms, for example:

History: The Battle of Hastings

The search then takes place in two phases; Firstly the local database is searched to locate the relevant search servers associated with history, the query is then translated into the required format for querying those systems; Secondly the found search servers are queried for information on The Battle of Hastings.

Librarians manually assign the words and aliases (or synonyms) that identify the subject associated with each search server. This degree of manual maintenance is considered acceptable because the frequency with which new search servers are added is very low in comparison with the frequency with which new Web pages are added to the individual databases.

6.2 *Automatic Classification*

An alternative to maintaining a separate database for each subject is to organise one database according to a classification scheme. The advantages of automatic classification [20] and document clustering[11] have long since been recognised by IR research. This is the approach supported by the manually maintained classified directories such as Yahoo. The result of classification is that documents sharing the same context are clustered into the same area of the classification scheme. Query results are generally restricted to one or two classes of the classification scheme and are therefore unlikely to inundate the user with irrelevant documents. Three projects that are exploring the advantages of automatic classification are detailed below:

TAPER

The Taxonomy And Path Enhanced Retrieval system[20] attempts to classify documents according to a hierarchical classification scheme. IR techniques are used to extract *signatures*[9] from documents based on significant terms and these are then compared with signatures representing each node of the classification hierarchy. Each node has a different context specific stop word list that is applied to the document signature as it is filtered down through the hierarchy. When a user queries the TAPER system, they are initially presented with a list of topic paths, rather than documents, this helps to focus the query to the most relevant areas of the classification hierarchy where subsequent relevant documents will be clustered.

Scorpion

The Scorpion project[21] of OCLC combines library science with IR techniques to provide an automated search engine that performs automatic classification. Dewey Decimal concepts are used as a knowledge base for automatic subject assignment.

Documents are used as queries to a database that contains information relating to the Dewey Decimal Classification scheme (DDC). The results from such queries identify the subject matter of each document. The Scorpion software stores information relating to each document in standard Dublin Core metadata files.

WWLib

The Wolverhampton Web Library (WWLib)[22] at the University of Wolverhampton is currently a manually maintained classified directory that classifies documents according to Dewey Decimal Classification (DDC). Design and implementation of an improved version of WWLib is underway which will support a number of fully automated features including a robot for resource location, automatic indexing software and an automatic classifier. Metadata will be stored in a standard compliant form for every resource that it locates.

Included in the metadata will be the DDC classmark that is generated as a result of automatic classification. The actual classification process, that is currently being developed, is similar to the TAPER system in that class representatives are generated for every node of the DDC classification hierarchy. These representatives comprise significant words and synonyms relating to that class. Document representatives are generated for each document encountered and these are then matched against the representatives of each node of the classification hierarchy. The

class representatives are broad at the top of the hierarchy and detailed at the bottom, having a similar filtering effect to the stop lists of the TAPER system. The DDC classmarks will then be used by the retrieval mechanism to cluster documents sharing similar concepts. Browsing of the classification hierarchy will also be possible.

6.3 The META Tag

The HTML META tag[23] provides a mechanism for improving the accuracy of existing automated tools by enabling Web authors to specify their own metadata. Most of the major search engines now support the meta tag, with the exception of Excite. The KEYWORDS and DESCRIPTION tags are the two most important META tags where search engines are concerned:

```
<HEAD>
<TITLE>Search Engines - An Evaluation</TITLE>
<META NAME="DESCRIPTION" CONTENT= "Information Resource
Discovery on the World Wide Web: An Evaluation of Tools and
Methodologies">
<META NAME="KEYWORDS" CONTENT="Search Engines, Classified
Directories, Meta Search Engines, Subject Specific gateways,
indexing,
Boolean Syntax">
</HEAD>
```

Placed in the HEAD element of an HTML page, these tags allow the author to specify index terms and textual descriptions of their documents. Automated tools then recognise these tags and use the information provided by the user to generate more accurate metadata. This process combines automation with human defined metadata and is generally thought to improve the accuracy of automated tools. Excite, however, object to the META tag on the grounds that it could be put to misuse. Authors could define inaccurate metadata, unrelated to the content of their page, to attract more accesses.

7 Conclusions

In general tools for information resource discovery on the World Wide Web suffer, to varying degrees, from being:

- incomplete - i.e. they do not cover all the information available and
- inaccurate - they contain information that is out of date, incorrect or classified incorrectly and they provide users with poor quality, often irrelevant, information.

It is evident that, in most cases, more effort is spent collecting new documents than verifying existing ones, hence the common occurrence of dead, out of date links in results. The trade off between completeness and accuracy is comparable to that between recall and precision - an age old dilemma for all information retrieval mechanisms.

Manually maintained classified directories, although intuitive to use and largely accurate, cover just a small fraction of the information available. The lack of automation makes them noticeably out of date and their human intervention can make their results opinionated and restricted.

Automated search engines, while being the most comprehensive tools in terms of Web coverage, are particularly prone to inaccuracy. Automatic analysis and categorisation is a complex task and although attempts made to cluster associated documents by concept are promising, there is still a tendency to skew the focus of a query making results much less accurate than those retrieved from a manually maintained, classified resource.

Meta search engines are a good starting point for beginners but deny the user access to the search options (boolean or otherwise) provided by each search engine. This problem could be eradicated by the development of a generic language, understood by the meta engine which could be translated into specific instructions for each search engine. Commands would be translated into the particular syntax each search engine required, providing a common front end to a set of specific search engines, each using a different syntax to implement essentially the same underlying model.

Geographically specific resources can help to avoid poor response times, cross Atlantic congestion and US biased information, but obviously, important, relevant information may not always be geographically local. The advantages of such resources are introduced by their limitations. Mirroring US resources avoids these limitations but reintroduces the problem of US bias.

The concept behind subject specific gateways is promising but only if a standard method, such as ROADS, is implemented universally so that they can all be linked together seamlessly. If they do not have some kind of central resource linking them all together they could ultimately be as difficult to locate as the individual pieces of information they store.

References

[1] P. Linder, *gopher-faq*, gopher://ftp.cac.psu.edu/00/internexus/GOPHER.FAQ, December 1992

[2] S. Foster, F. Barrie, *veronica-faq*, gopher://veronica.scs.unr.edu, June 1993

[3] O. A. McBryan, *GENVL and WWWW: Tools for Taming the Web*, http://www.cs.colorado.edu/home/mcbryan/mypapers/www94.ps From proceedings of the First International World Wide Web Conference, ed. O. Nierstrasz, CERN, Geneva, May 1994

[4] L. Lindop, M. Sriskandarajah, M. Williams, M. Bracken, M. Cadden, A. Dabbs, W. Gallagher, *Catching Sites*, PC Magazine, Volume 6, Issue 2, P109-153, February 1997

[5] M. Koster, *The Web Robots Page*, http://info.webcrawler.com/mak/projects/robots/

[6] M. Day, A. Powell, *Metadata*, http://www.ukoln.ac.uk/metadata, October 1997

[7] J. Kirriemuir, *Resource Organisation And Discovery in Subject-based services (ROADS)*, http://ukoln.bath.ac.uk/roads/intro.html, October 1996

[8] G. Salton, M.J. McGill, *Introduction To Modern Information Retrieval*, McGraw Hill, ISBN 0-07-066526-5, 1983

[9] G. Kowalski, *Information Retrieval Systems Theory and Implementation*, Kluwer, ISBN 0-7923-9926-9, 1997

[10] Excite Inc., *Information Retrieval Technology and Intelligent Concept Extraction Searching*, http://www.excite.com/ice/tech.html, 1996

[11] R. N. Oddy, S. E. Robertson, C. J. van Rijsbergen, P. W. Williams, *Information Retrieval Research*, Butterworths, ISBN 0-408-10775-8, Section 4, P35, 1981

[12] Muscat, *Muscat*, http://www.muscat.com, 1997

[13] M.F. Porter, *An Algorithm for Suffix Stripping*, Program, 1980

[14] W. B. Frakes, R. Baeza-Yates, *Information Retrieval data Structures and Algorithms*, Prentice Hall, 1992

[15] D. Beckett, *IAFA Templates in use as Internet Metadata*, From the proceedings of the 4th International World Wide Web Conference, http://www.w3.org/conferences/www4/papers/52/, Boston, Massachusetts, December 1995

[16] S. Weibel, J. Godby, E. Miller, R. Daniel, *OCLC/NCSA Metadata Workshop Report*, http://www.oclc.org:5046/conferences/metadat/dublin_core_report.html, March 1995

[17] U. Manber, *The Search Broker*, http://sb.cs.arizona.edu/sb/paper.html, July 1997

[18] U. Manber, B. Gopal, S. Wu, *GLIMPSE: A tool to search entire file systems*, http://glimpse.cs.arizona.edu/

[19] M. Bowman, P. Danzig, D. Hardy, U. Manber, M. Schwartz, D. Wessels, *The Harvest Information Discovery and Access System,* http://harvest.transarc.com, 1996

[20] S. Chakrabarti, B. Dom, R. Agrawal, P. Raghavan, *Using taxonomy, discriminants, and signatures for navigating in text databases*, Proceedings of the 23rd VLDB Conference, Athens, Greece, 1997

[21] R. Thompson, K. Shafer, D. Vizine-Goetz, *Evaluating Dewey Concepts as a Knowledge Base for Automatic Subject Assignment*, http://orc.rsch.oclc.org:6109/eval_de.html, February 1997

[22] P. Burden, *The UK Web Library - WWLib*, http://www.scit.wlv.ac.uk/wwlib/, 1996

[23] M. C. Miller, *HTML Writers Guild Using Meta for Search Engines FAQ*, http://www.hwg.org/faqs/metafaq.html, January 1997

Table 1 - Classified Directories

Name	URL	Search Facility	Initial Categories	Site Reviews	Advanced Search
Galaxy	http://www.einet.net/galaxy.html	Y	11	N	Y
Identify	http://www.identify.com/	Y	14	N	N
I-explorer	http://www.i-explorer.com/	Y	21	N	Y
Link Center	http://www.linkcenter.com/	Y	14	N	N
Link Monster	http://www.linkmonster.com/	Y	27	N	N
Magellan	http:/www.mckinley.com/	Y	15	Y	Y
Nerd World	http://www.nerdworld.com/	Y	23	Y	N
Search.com	http://www.search.com/	Y	25	N	N
Yellow Pages	http://www.mcp.com/nrp/wwwyp/	Y	101	Y	N
Yahoo	http://www.yahoo.com/	Y	14	Y	Y

Table 2 - Search Engine Evolution

Name	Date
GENVL	1993
WWW Worm	1993
Galaxy	1993
Yahoo	1994
Lycos	May 1994
Infoseek	Early 1995
Excite	Late 1995
AltaVista	Dec 1995
HotBot	1996

Table 3 - Automated Search Engines

Name	URL	Corpus Size	Fully Indexed	Boolean Search	Classified Directory	Description	Relevance Score	Date	Proximity Search	META support
Alta Vista	http://www.altavista.digital.com/	30,000000	Y	Y	N	Y	N	Y	Y	Y
Excite	http://www.excite.com/	50,000,000	Y	Y	Y	Y	Y	N	Y	Y
HotBot	http://www.hotbot.com/	54,000,000	Y	Y	Y	Y	Y	Y	N	Y
Infoseek	http://www.infoseek.com/	80,000,000	Y	N	Y	Y	Y	N	N	Y
Lycos	http://www.lycos.com/	66,557,000	N	Y	Y	Y	Y	N	Y	Y
OpenText	http://search.opentext.com/	?	Y	Y	N	Y	Y	N	Y	N
WebCrawler	http://webcrawler.com/	?	Y	Y	Y	Y	Y	N	N	Y

Table 4 - Meta Search Engines

NAME	URL	RESULTS COLLATED?	NUMBER OF SEARCII ENGINES QUERIED
All 4 One	http://all4one.com/	N	4
Cyber411	http://cyber411.com/	Y	6
Dogpile	http://www.dogpile.com/	Y	14
Highway61	http://www.highway61.com/	Y	4
Internet Sleuth	http://www.isleuth.com/	Y	6
MetaCrawler	http://www.metacrawler.com	Y	6
Metasearch	http://metasearch.com/	N	6
Pro Fusion	http://www.designlab.ukans.edu/profusion/	Y	6
Savvy Search	http://guaraldi.cs.colostate.edu:2000/form	Y	11
StartingPoint	http://www.stpt.com/	N	160

Table 5 - Geographically Specific Resources

NAME	URL	COUNTRY	TYPE
ANANZI	http://www.anazi.co.za/	South Africa	Search Engine
ANZWERS	http://www.anzwers.com.au/	Australia & New Zealand	Search Engine
Channel Hong Kong	http://www.chkg.com/	Hong Kong	Search Engine
Euroferret	http:/www.euroferret.com/	Europe	Search Engine
Kolibri	http://www.kolibri.de/	Germany	Search Engine
Search.NL	http://www.search.NL/	Holland	Search Engine
Swiss Search	http://search.ch/	Switzerland	Search Engine
TechnoFind	http://www2.technofind.com.sg/tf/	Singapore	Search Engine
UK Index	http://www.ukindex.co.uk/uksearch.html	UK	Classified Directory
UK Plus	http://www.ukplus.co.uk/	UK	Classified Directory
UKSearch	http://www.uksearch.com/	UK	Search Engine
The UK Web Pages	http://www.neosoft.com/_dlgates/uk /ukgeneral.html	UK	Classified Directory
YELL	http://www.yell.co.uk/	UK	Classified Directory
ZZZ	http://www.zzz.ee/otsi/index_en.html	Estonia	Search Engine

Table 6 - Subject Specific Gateways

NAME	URL	SUBJECT	TYPE	LOCATION
1.2.1.2.	http://www.1212.com/	Music	Classified Directory	France
Achoo	http://www.achoo.com/	Healthcare	Classified Directory	Canada
ADAM*	http://www.adam.co.uk/	Architecture, Design And Media	Classified Directory	UK
Aqueous	http://www.aqueous.com/	Water Related	Search Engine	?
ASE	http://www.uni-karlsruche.de /~un9v/atm/ase.html	Airport Search Engine	Search Engine	Germany
BizAds Business locator	http://bizads.2cowherd.net/	Businesses	Search Engine	USA
CampSearch	http://www.campsearch.com/	Summer Camps	Search Engine	USA
Cinemachine	http://www.cinemachine.com/	Film Reviews	Search Engine	?
Computer ESP	http://www.uvision.com /search.html	Computer Companies and Products	Classified Directory	USA
EEVL*	http://eevl.ac.uk/	Edinburgh Engineering Virtual Library	Classified Directory	UK
1st Global Directory	http://www.123link.com/	Business Products and Services	Classified Directory	USA
Motherload	http://www.cosmix.com /motherload/	Web Directories and Search Engines!	Classified Directory	USA
NetMall	http://www.netmall.com/	Goods and Services	Classified Directory	USA
OMNI*	http://omni.ac.uk/	Organising Medical Networked information	Classified Directory	UK
SHAREWARE.COM	http://www.shareware.com/	Software	Search Engine	USA
SOSIG*	http://sosig.esrc.bris.ac.uk/	Social Sciences Information gateway	Classified Directory	UK
Sports Directory	http://www.sport-hq.com/	Sport	Classified Directory	USA

* These initiatives utilise ROADS (Resource Organisation And Discovery System) technology.